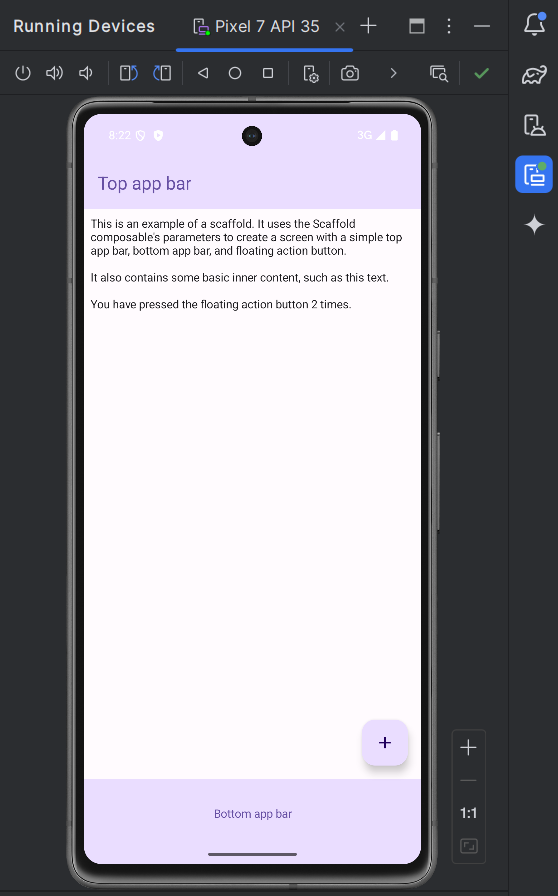
**Sample 1**

****

package com.example.scaffold\_sample  
  
import android.os.Bundle  
import androidx.activity.ComponentActivity  
import androidx.activity.compose.setContent  
import androidx.activity.enableEdgeToEdge  
import androidx.compose.foundation.layout.Arrangement  
import androidx.compose.foundation.layout.Column  
import androidx.compose.foundation.layout.fillMaxSize  
import androidx.compose.foundation.layout.fillMaxWidth  
import androidx.compose.foundation.layout.padding  
import androidx.compose.material.icons.Icons  
import androidx.compose.material.icons.filled.*Add*import androidx.compose.material3.BottomAppBar  
import androidx.compose.material3.ExperimentalMaterial3Api  
import androidx.compose.material3.FloatingActionButton  
import androidx.compose.material3.Icon  
import androidx.compose.material3.MaterialTheme  
import androidx.compose.material3.Scaffold  
import androidx.compose.material3.Text  
import androidx.compose.material3.TopAppBar  
import androidx.compose.material3.TopAppBarDefaults.topAppBarColors  
import androidx.compose.runtime.Composable  
import androidx.compose.runtime.getValue  
import androidx.compose.runtime.mutableIntStateOf  
import androidx.compose.runtime.remember  
import androidx.compose.runtime.setValue  
import androidx.compose.ui.Modifier  
import androidx.compose.ui.text.style.TextAlign  
import androidx.compose.ui.tooling.preview.Preview  
import androidx.compose.ui.unit.dp  
import com.example.scaffold\_sample.ui.theme.Scaffold\_sampleTheme

Class MainActivity : ComponentActivity() {  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 *enableEdgeToEdge*()  
 *setContent* **{** ScaffoldExample()  
 **}** }  
}

OptIn(ExperimentalMaterial3Api::class)  
@Composable  
fun ScaffoldExample() {  
 var presses by remember **{** *mutableIntStateOf*(0) **}** Scaffold(

topBar = **{** TopAppBar(  
 colors = topAppBarColors(  
 containerColor = MaterialTheme.colorScheme.primaryContainer,  
 titleContentColor = MaterialTheme.colorScheme.primary,  
 ),  
 title = **{** Text("Top app bar")  
 **}** )  
 **}**,

bottomBar = **{** BottomAppBar(  
 containerColor = MaterialTheme.colorScheme.primaryContainer,  
 contentColor = MaterialTheme.colorScheme.primary,  
 ) **{** Text(  
 modifier = Modifier  
 .*fillMaxWidth*(),  
 textAlign = TextAlign.Center,  
 text = "Bottom app bar",  
 )  
 **}  
 }**,

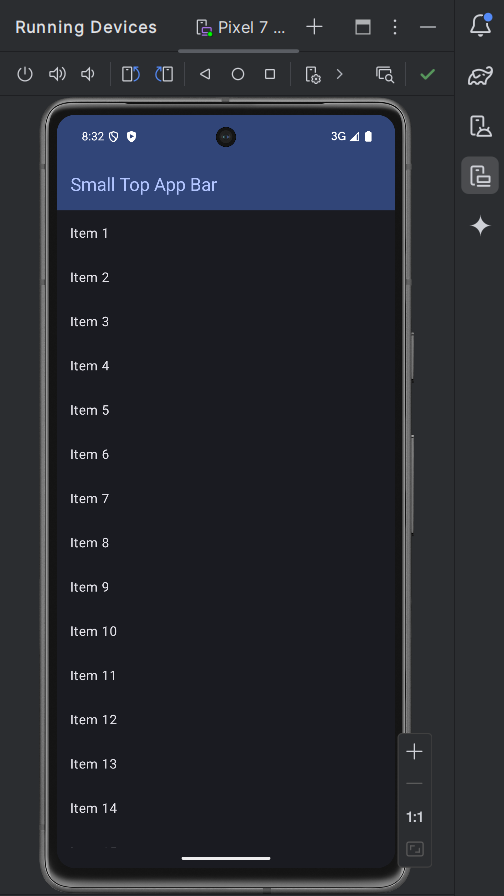
floatingActionButton = **{** FloatingActionButton(onClick = **{** presses++ **}**) **{** Icon(Icons.Default.*Add*, contentDescription = "Add")  
 **}  
 }** ) //Scaffold closing bracket

**{** innerPadding **->**

Column(  
 modifier = Modifier  
 .*padding*(innerPadding),  
 verticalArrangement = Arrangement.spacedBy(16.*dp*),  
 )

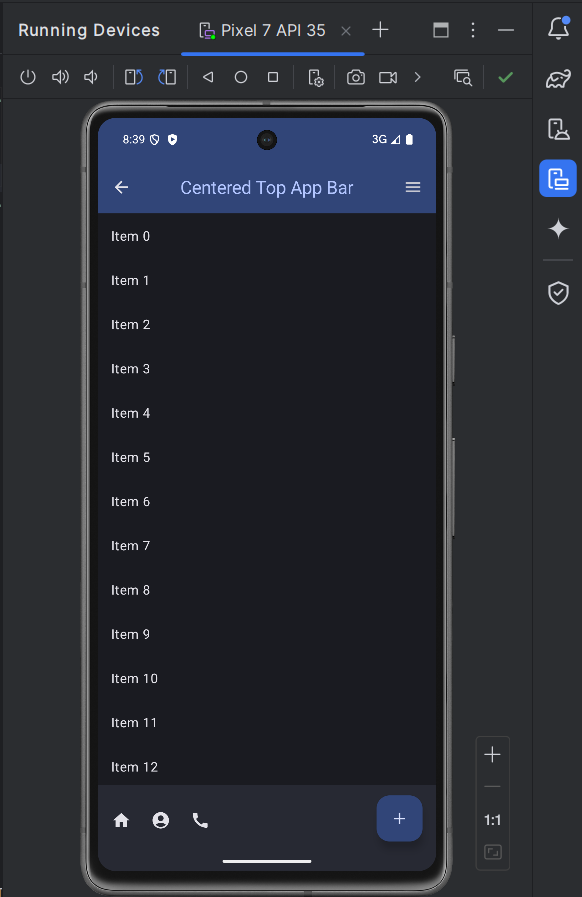
**{** Text(  
 modifier = Modifier.*padding*(8.*dp*),  
 text =  
 """  
 This is an example of a scaffold. It uses the Scaffold composable's parameters to create a screen with a simple top app bar, bottom app bar, and floating action button.  
  
 It also contains some basic inner content, such as this text.  
  
 You have pressed the floating action button $presses times.  
 """.*trimIndent*(),  
 )  
 **}  
 }**}

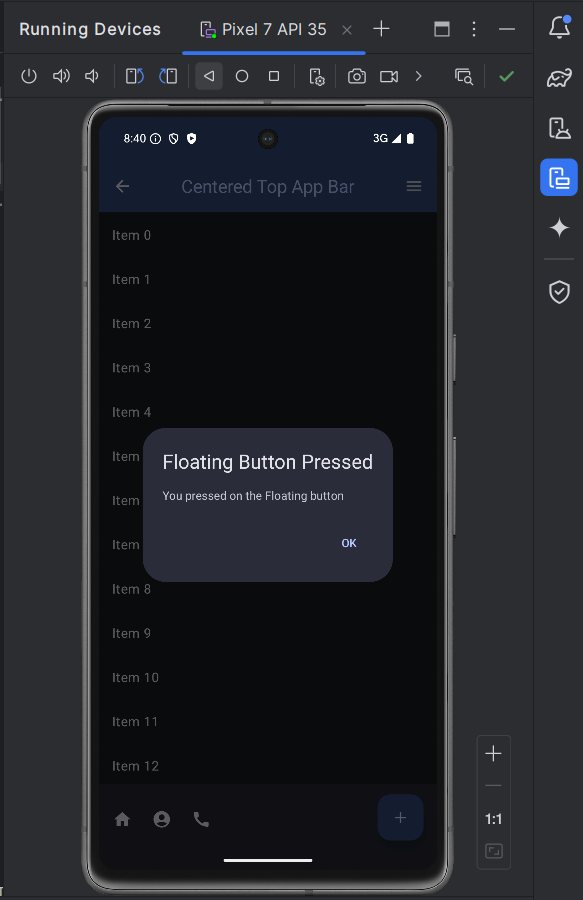
**Sample 2**

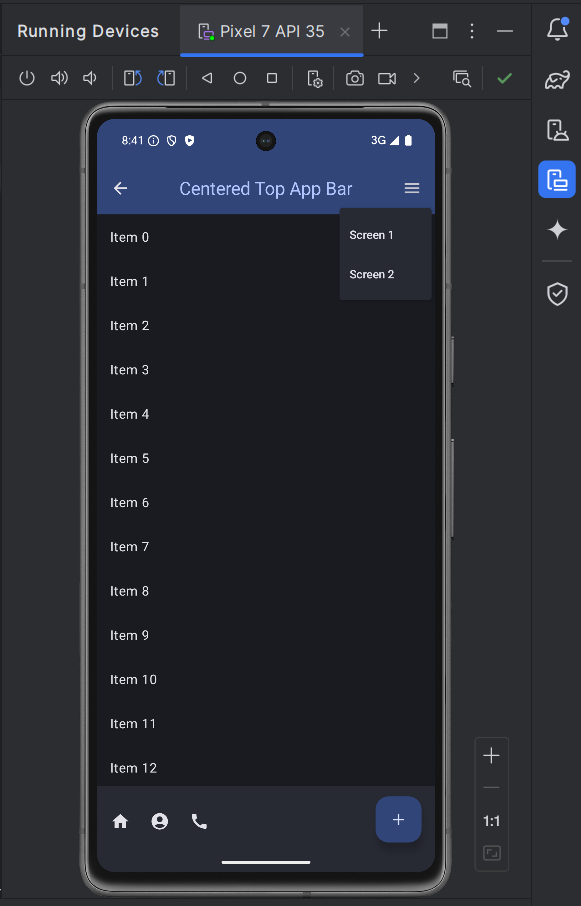


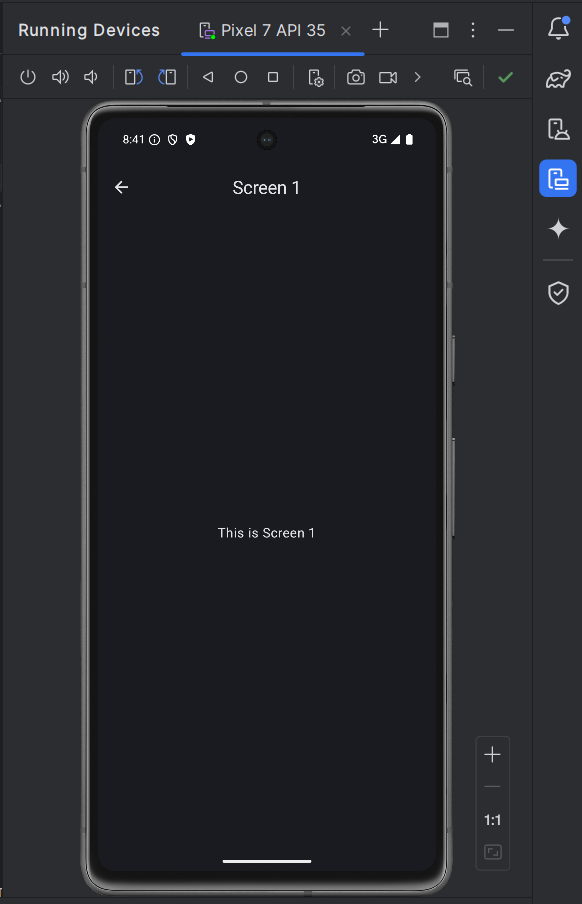
package com.example.appbars  
  
import android.os.Bundle  
import androidx.activity.ComponentActivity  
import androidx.activity.compose.setContent  
import androidx.activity.enableEdgeToEdge  
import androidx.compose.foundation.layout.PaddingValues  
import androidx.compose.foundation.layout.fillMaxSize  
import androidx.compose.foundation.layout.padding  
import androidx.compose.material3.ExperimentalMaterial3Api  
import androidx.compose.material3.MaterialTheme  
import androidx.compose.material3.Scaffold  
import androidx.compose.material3.Text  
import androidx.compose.material3.TopAppBar  
import androidx.compose.material3.TopAppBarDefaults  
import androidx.compose.runtime.Composable  
import androidx.compose.ui.Modifier  
import androidx.compose.ui.tooling.preview.Preview  
import androidx.compose.foundation.lazy.LazyColumn  
import androidx.compose.foundation.lazy.items  
import androidx.compose.ui.unit.dp  
import com.example.appbars.ui.theme.AppBarsTheme  
  
class MainActivity : ComponentActivity() {  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 *enableEdgeToEdge*()  
 *setContent* **{** AppBarsTheme **{** SmallTopAppBarExample()  
 **}  
 }** }  
}  
  
@OptIn(ExperimentalMaterial3Api::class)  
@Composable  
fun SmallTopAppBarExample() {  
 Scaffold(  
 topBar = **{** TopAppBar(  
 colors = TopAppBarDefaults.topAppBarColors(  
 containerColor = MaterialTheme.colorScheme.primaryContainer,  
 titleContentColor = MaterialTheme.colorScheme.primary,  
 ),  
 title = **{** Text("Small Top App Bar")  
 **}** )  
 **}**,  
 ) **{** innerPadding **->** ScrollContent(innerPadding)  
 **}**}  
  
@Composable  
fun ScrollContent(innerPadding: PaddingValues) {  
 val itemsList = *listOf*(  
 "Item 1",  
 "Item 2",  
 "Item 3",  
 "Item 4",  
 "Item 5",  
 "Item 6",  
 "Item 7",  
 "Item 8",  
 "Item 9",  
 "Item 10",  
 "Item 11",  
 "Item 12",  
 "Item 13",  
 "Item 14",  
 "Item 15",  
 "Item 16",  
 "Item 17",  
 "Item 18",  
 "Item 19",  
 "Item 20",  
 "Item 21",  
 "Item 22",  
 "Item 23",  
 "Item 24",  
 "Item 25",  
 "Item 26",  
 "Item 27",  
 "Item 28",  
 "Item 29",  
 "Item 30",  
 "Item 31",  
 "Item 32",  
 "Item 33",  
 "Item 34",  
 "Item 35",  
 "Item 36",  
 "Item 37",  
 "Item 38"  
 )  
  
 LazyColumn(  
 modifier = Modifier  
 .*fillMaxSize*()  
 .*padding*(innerPadding)  
 ) **{** *items*(itemsList) **{** item **->** Text(text = item, modifier = Modifier.*padding*(16.*dp*))  
 **}  
 }**}  
  
@Preview(showBackground = true)  
@Composable  
fun DefaultPreview() {  
 AppBarsTheme **{** SmallTopAppBarExample()  
 **}**}

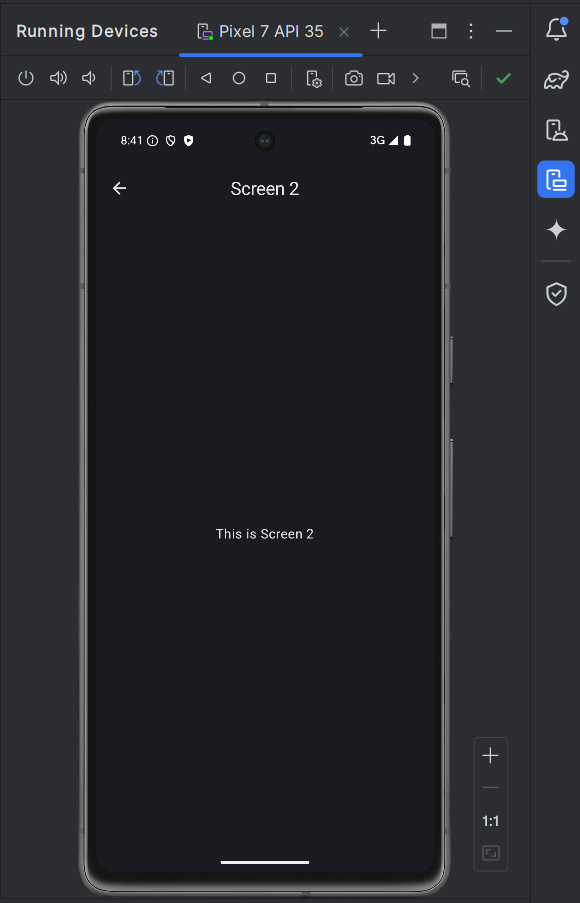
**Sample 3**











package com.example.appbarcenteraligned  
  
import android.os.Bundle  
import androidx.activity.ComponentActivity  
import androidx.activity.compose.setContent  
import androidx.activity.enableEdgeToEdge  
import androidx.compose.foundation.layout.Arrangement  
import androidx.compose.foundation.layout.Box  
import androidx.compose.foundation.layout.Column  
import androidx.compose.foundation.layout.PaddingValues  
import androidx.compose.foundation.layout.fillMaxSize  
import androidx.compose.foundation.layout.fillMaxWidth  
import androidx.compose.foundation.layout.padding  
import androidx.compose.foundation.rememberScrollState  
import androidx.compose.foundation.verticalScroll  
import androidx.compose.material.icons.Icons  
import androidx.compose.material.icons.filled.\*  
import androidx.compose.material3.\*  
import androidx.compose.runtime.Composable  
import androidx.compose.runtime.mutableStateOf  
import androidx.compose.runtime.remember  
import androidx.compose.ui.Alignment  
import androidx.compose.ui.Modifier  
import androidx.compose.ui.input.nestedscroll.nestedScroll  
import androidx.compose.ui.text.style.TextOverflow  
import androidx.compose.ui.unit.dp  
import androidx.navigation.NavHostController  
import androidx.navigation.compose.NavHost  
import androidx.navigation.compose.composable  
import androidx.navigation.compose.rememberNavController  
import com.example.appbarcenteraligned.ui.theme.AppBarCenterAlignedTheme  
  
class MainActivity : ComponentActivity() {  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 *enableEdgeToEdge*()  
 *setContent* **{** AppBarCenterAlignedTheme **{** val navController = rememberNavController()  
 NavigationComponent(navController)  
 **}  
 }** }  
}  
  
@OptIn(ExperimentalMaterial3Api::class)  
@Composable  
fun CenterAlignedTopAppBarExample(navController: NavHostController) {  
 val scrollBehavior = TopAppBarDefaults.pinnedScrollBehavior(rememberTopAppBarState())  
 val expanded = remember **{** *mutableStateOf*(false) **}** val showDialog = remember **{** *mutableStateOf*(false) **}** if (showDialog.value) {  
 AlertDialog(  
 onDismissRequest = **{** showDialog.value = false **}**,  
 confirmButton = **{** TextButton(onClick = **{** showDialog.value = false **}**) **{** Text("OK")  
 **}  
 }**,  
 title = **{** Text("Floating Button Pressed") **}**,  
 text = **{** Text("You pressed on the Floating button") **}** )  
 }  
  
 Scaffold(  
 modifier = Modifier.*nestedScroll*(scrollBehavior.nestedScrollConnection),

topBar = **{** CenterAlignedTopAppBar(  
 colors = TopAppBarDefaults.topAppBarColors(  
 containerColor = MaterialTheme.colorScheme.primaryContainer,  
 titleContentColor = MaterialTheme.colorScheme.primary,

),

title = **{** Text(  
 "Centered Top App Bar",  
 maxLines = 1,  
 overflow = TextOverflow.Ellipsis  
 )  
 **}**,

navigationIcon = **{** IconButton(onClick = **{** navController.popBackStack() **}**) **{** Icon(  
 imageVector = Icons.Filled.*ArrowBack*,  
 contentDescription = "Back"  
 )  
 **}  
 }**,

actions = **{**

IconButton(onClick = **{** expanded.value = !expanded.value **}**) **{** Icon(  
 imageVector = Icons.Filled.*Menu*,  
 contentDescription = "Menu"  
 )  
 **}**

DropdownMenu(  
 expanded = expanded.value,  
 onDismissRequest = **{** expanded.value = false **}** ) **{** DropdownMenuItem(  
 text = **{** Text("Screen 1") **}**,  
 onClick = **{** expanded.value = false  
 navController.navigate("screen1")  
 **}** )  
 DropdownMenuItem(  
 text = **{** Text("Screen 2") **}**,  
 onClick = **{** expanded.value = false  
 navController.navigate("screen2")  
 **}** )  
 **}  
 }**,

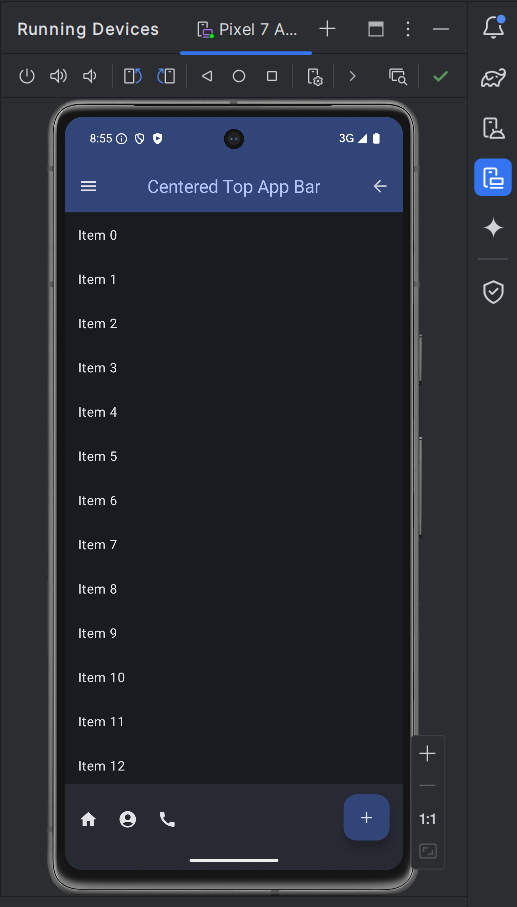
scrollBehavior = scrollBehavior,  
 )  
 **}**,

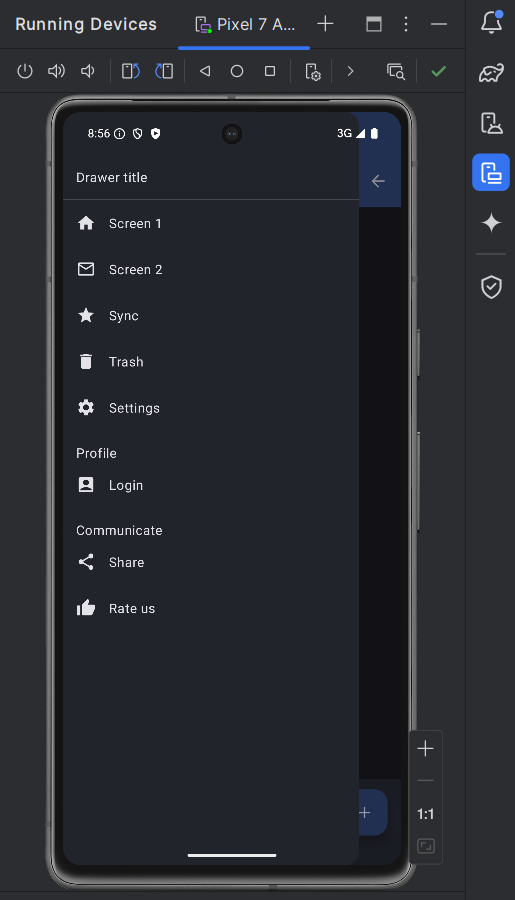
bottomBar = **{**

BottomAppBar(

actions = **{** IconButton(onClick = **{** /\* Handle bottom button 1 click \*/ **}**) **{** Icon(imageVector = Icons.Filled.*Home*, contentDescription = "Button 1")  
 **}** IconButton(onClick = **{** /\* Handle bottom button 2 click \*/ **}**) **{** Icon(imageVector = Icons.Filled.*AccountCircle*, contentDescription = "Button 2")  
 **}** IconButton(onClick = **{** /\* Handle bottom button 3 click \*/ **}**) **{** Icon(imageVector = Icons.Filled.*Call*, contentDescription = "Button 3")  
 **}  
 }**,  
 floatingActionButton = **{** FloatingActionButton(onClick = **{** showDialog.value = true **}**) **{** Icon(imageVector = Icons.Filled.*Add*, contentDescription = "Floating Button")  
 **}  
 }** )  
 **}**,  
 floatingActionButtonPosition = FabPosition.End,  
 ) **{** innerPadding **->** MyScrollContent(innerPadding)  
 **}**}  
  
@Composable  
fun MyScrollContent(padding: PaddingValues) {  
 // Create a scroll state  
 val scrollState = rememberScrollState()  
  
 // Create a Column that will hold the scrollable content  
 Column(  
 modifier = Modifier  
 .*padding*(padding) // Apply the padding received from the Scaffold  
 .*verticalScroll*(scrollState) // Make the Column scrollable  
 .*fillMaxSize*() // Make the Column fill the available space  
 ) **{** // Add some content to the Column  
 *repeat*(20) **{** index **->** Text(  
 text = "Item $index",  
 modifier = Modifier  
 .*padding*(16.*dp*)  
 .*fillMaxWidth*()  
 )  
 **}  
 }**}  
  
@Composable  
fun NavigationComponent(navController: NavHostController) {  
 NavHost(navController, startDestination = "main") **{** *composable*("main") **{** CenterAlignedTopAppBarExample(navController) **}** *composable*("screen1") **{** Screen1(navController) **}** *composable*("screen2") **{** Screen2(navController) **}  
 }**}  
  
@OptIn(ExperimentalMaterial3Api::class)  
@Composable  
fun Screen1(navController: NavHostController) {  
 Scaffold(  
 topBar = **{** CenterAlignedTopAppBar(  
 title = **{** Text("Screen 1")  
 **}**,  
 navigationIcon = **{** IconButton(onClick = **{** navController.navigate("main") **}**) **{** Icon(  
 imageVector = Icons.Filled.*ArrowBack*,  
 contentDescription = "Back"  
 )  
 **}  
 }** )  
 **}** ) **{** Column(  
 modifier = Modifier  
 .*fillMaxSize*()  
 .*padding*(**it**),  
 horizontalAlignment = Alignment.CenterHorizontally,  
 verticalArrangement = Arrangement.Center  
 ) **{** Text(text = "This is Screen 1")  
 **}  
 }**}  
  
@OptIn(ExperimentalMaterial3Api::class)  
@Composable  
fun Screen2(navController: NavHostController) {  
 Scaffold(  
 topBar = **{** CenterAlignedTopAppBar(  
 title = **{** Text("Screen 2")  
 **}**,  
 navigationIcon = **{** IconButton(onClick = **{** navController.navigate("main") **}**) **{** Icon(  
 imageVector = Icons.Filled.*ArrowBack*,  
 contentDescription = "Back"  
 )  
 **}  
 }** )  
 **}** ) **{** Column(  
 modifier = Modifier  
 .*fillMaxSize*()  
 .*padding*(**it**),  
 horizontalAlignment = Alignment.CenterHorizontally,  
 verticalArrangement = Arrangement.Center  
 ) **{** Text(text = "This is Screen 2")  
 **}  
 }**}

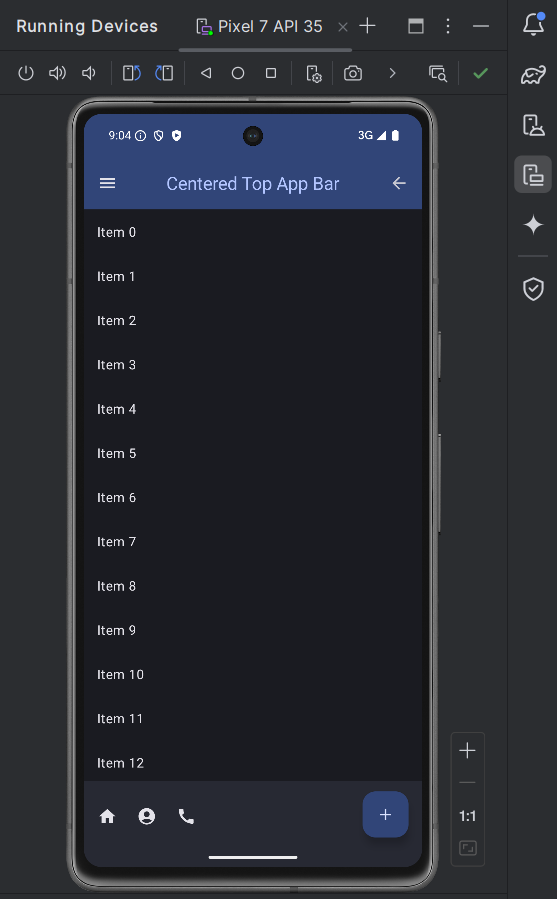
**Sample 4**

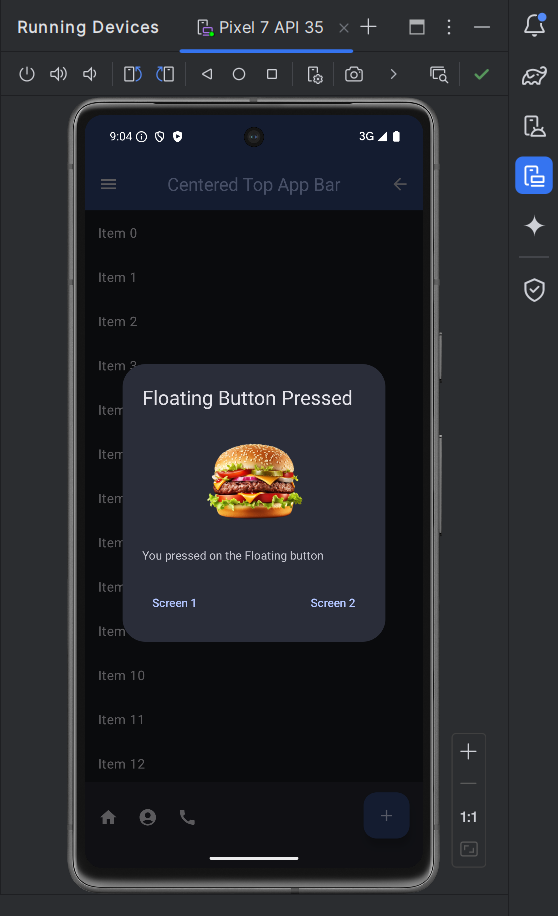


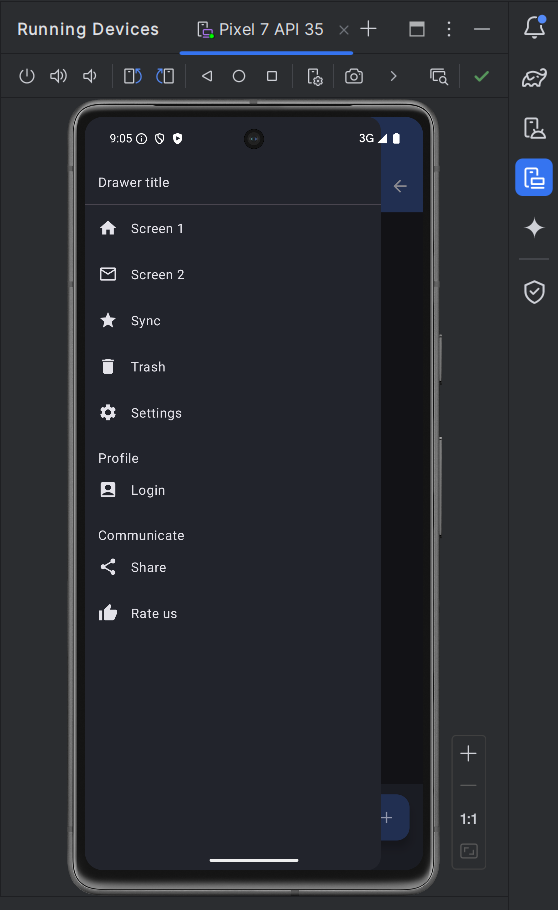


package com.example.appbarcenteraligned  
  
import android.os.Bundle  
import androidx.activity.ComponentActivity  
import androidx.activity.compose.setContent  
import androidx.activity.enableEdgeToEdge  
import androidx.compose.foundation.Image  
import androidx.compose.foundation.background  
import androidx.compose.foundation.clickable  
import androidx.compose.foundation.layout.\*  
import androidx.compose.foundation.rememberScrollState  
import androidx.compose.foundation.shape.*CircleShape*import androidx.compose.foundation.verticalScroll  
import androidx.compose.material.icons.Icons  
import androidx.compose.material.icons.filled.\*  
import androidx.compose.material3.\*  
import androidx.compose.runtime.\*  
import androidx.compose.ui.Alignment  
import androidx.compose.ui.Modifier  
import androidx.compose.ui.draw.clip  
import androidx.compose.ui.graphics.Color  
import androidx.compose.ui.graphics.vector.ImageVector  
import androidx.compose.ui.input.nestedscroll.nestedScroll  
import androidx.compose.ui.res.painterResource  
import androidx.compose.ui.text.font.FontWeight  
import androidx.compose.ui.text.style.TextOverflow  
import androidx.compose.ui.unit.dp  
import androidx.navigation.NavHostController  
import androidx.navigation.compose.NavHost  
import androidx.navigation.compose.composable  
import androidx.navigation.compose.rememberNavController  
import com.example.appbarcenteraligned.ui.theme.AppBarCenterAlignedTheme  
import kotlinx.coroutines.CoroutineScope  
import kotlinx.coroutines.launch  
  
class MainActivity : ComponentActivity() {  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 *enableEdgeToEdge*()  
 *setContent* **{** AppBarCenterAlignedTheme **{** val navController = rememberNavController()  
 NavigationComponent(navController)  
 **}  
 }** }  
}  
  
@OptIn(ExperimentalMaterial3Api::class)  
@Composable  
fun CenterAlignedTopAppBarExample(navController: NavHostController) {  
 val drawerState = rememberDrawerState(DrawerValue.*Closed*)  
 val coroutineScope = rememberCoroutineScope()  
 val scrollBehavior = TopAppBarDefaults.pinnedScrollBehavior(rememberTopAppBarState())  
 val showDialog = remember **{** *mutableStateOf*(false) **}** if (showDialog.value) {  
 AlertDialog(  
 onDismissRequest = **{** showDialog.value = false **}**,  
 confirmButton = **{** TextButton(onClick = **{** showDialog.value = false **}**) **{** Text("OK")  
 **}  
 }**,  
 title = **{** Text("Floating Button Pressed") **}**,  
 text = **{** Text("You pressed on the Floating button") **}** )  
 }  
  
 ModalNavigationDrawer(  
 drawerState = drawerState,  
 drawerContent = **{** DrawerContent(navController, drawerState, coroutineScope)  
 **}** ) **{** Scaffold(  
 modifier = Modifier.*nestedScroll*(scrollBehavior.nestedScrollConnection),  
 topBar = **{** CenterAlignedTopAppBar(  
 colors = TopAppBarDefaults.topAppBarColors(  
 containerColor = MaterialTheme.colorScheme.primaryContainer,  
 titleContentColor = MaterialTheme.colorScheme.primary,  
 ),  
 title = **{** Text(  
 "Centered Top App Bar",  
 maxLines = 1,  
 overflow = TextOverflow.Ellipsis  
 )  
 **}**,  
 navigationIcon = **{** IconButton(onClick = **{** coroutineScope.*launch* **{** drawerState.open() **} }**) **{** Icon(  
 imageVector = Icons.Filled.*Menu*,  
 contentDescription = "Menu"  
 )  
 **}  
 }**,  
 actions = **{** IconButton(onClick = **{** navController.popBackStack() **}**) **{** Icon(  
 imageVector = Icons.Filled.*ArrowBack*,  
 contentDescription = "Back"  
 )  
 **}  
 }**,  
 scrollBehavior = scrollBehavior,  
 )  
 **}**,  
 bottomBar = **{** BottomAppBar(  
 actions = **{** IconButton(onClick = **{** /\* Handle bottom button 1 click \*/ **}**) **{** Icon(imageVector = Icons.Filled.*Home*, contentDescription = "Button 1")  
 **}** IconButton(onClick = **{** /\* Handle bottom button 2 click \*/ **}**) **{** Icon(imageVector = Icons.Filled.*AccountCircle*, contentDescription = "Button 2")  
 **}** IconButton(onClick = **{** /\* Handle bottom button 3 click \*/ **}**) **{** Icon(imageVector = Icons.Filled.*Call*, contentDescription = "Button 3")  
 **}  
 }**,  
 floatingActionButton = **{** FloatingActionButton(onClick = **{** showDialog.value = true **}**) **{** Icon(imageVector = Icons.Filled.*Add*, contentDescription = "Floating Button")  
 **}  
 }** )  
 **}**,  
 floatingActionButtonPosition = FabPosition.End,  
 ) **{** innerPadding **->** MyScrollContent(innerPadding)  
 **}  
 }**}  
  
@Composable  
fun DrawerContent(navController: NavHostController, drawerState: DrawerState, coroutineScope: CoroutineScope) {  
 ModalDrawerSheet **{** Text("Drawer title", modifier = Modifier.*padding*(16.*dp*))  
 Divider()  
 DrawerItem(icon = Icons.Filled.*Home*, label = "Screen 1", onClick = **{** coroutineScope.*launch* **{** drawerState.close() **}** navController.navigate("screen1")  
 **}**)  
 DrawerItem(icon = Icons.Filled.*MailOutline*, label = "Screen 2", onClick = **{** coroutineScope.*launch* **{** drawerState.close() **}** navController.navigate("screen2")  
 **}**)  
 DrawerItem(icon = Icons.Filled.*Star*, label = "Sync", onClick = **{** coroutineScope.*launch* **{** drawerState.close() **}** navController.navigate("sync")  
 **}**)  
 DrawerItem(icon = Icons.Filled.*Delete*, label = "Trash", onClick = **{** coroutineScope.*launch* **{** drawerState.close() **}** navController.navigate("trash")  
 **}**)  
 DrawerItem(icon = Icons.Filled.*Settings*, label = "Settings", onClick = **{** coroutineScope.*launch* **{** drawerState.close() **}** navController.navigate("settings")  
 **}**)  
  
 // Profile and Communicate Sections  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
 Text("Profile", modifier = Modifier.*padding*(start = 16.*dp*))  
 DrawerItem(icon = Icons.Filled.*AccountBox*, label = "Login", onClick = **{** coroutineScope.*launch* **{** drawerState.close() **}** navController.navigate("login")  
 **}**)  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
 Text("Communicate", modifier = Modifier.*padding*(start = 16.*dp*))  
 DrawerItem(icon = Icons.Filled.*Share*, label = "Share", onClick = **{** coroutineScope.*launch* **{** drawerState.close() **}** navController.navigate("share")  
 **}**)  
 DrawerItem(icon = Icons.Filled.*ThumbUp*, label = "Rate us", onClick = **{** coroutineScope.*launch* **{** drawerState.close() **}** navController.navigate("rate\_us")  
 **}**)  
 // ...other drawer items  
 **}**}  
  
@Composable  
fun DrawerItem(icon: ImageVector, label: String, onClick: () -> Unit) {  
 Row(  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*clickable*(onClick = onClick)  
 .*padding*(16.*dp*),  
 verticalAlignment = Alignment.CenterVertically  
 ) **{** Icon(icon, contentDescription = label)  
 Spacer(modifier = Modifier.*width*(16.*dp*))  
 Text(label)  
 **}**}  
  
@Composable  
fun MyScrollContent(padding: PaddingValues) {  
 val scrollState = rememberScrollState()  
 Column(  
 modifier = Modifier  
 .*padding*(padding)  
 .*verticalScroll*(scrollState)  
 .*fillMaxSize*()  
 ) **{** *repeat*(20) **{** index **->** Text(  
 text = "Item $index",  
 modifier = Modifier  
 .*padding*(16.*dp*)  
 .*fillMaxWidth*()  
 )  
 **}  
 }**}  
  
@Composable  
fun NavigationComponent(navController: NavHostController) {  
 NavHost(navController, startDestination = "main") **{** *composable*("main") **{** CenterAlignedTopAppBarExample(navController) **}** *composable*("screen1") **{** Screen1(navController) **}** *composable*("screen2") **{** Screen2(navController) **}** *composable*("message") **{** ScreenMessage(navController) **}** *composable*("sync") **{** ScreenSync(navController) **}** *composable*("trash") **{** ScreenTrash(navController) **}** *composable*("settings") **{** ScreenSettings(navController) **}** *composable*("login") **{** ScreenLogin(navController) **}** *composable*("share") **{** ScreenShare(navController) **}** *composable*("rate\_us") **{** ScreenRateUs(navController) **}  
 }**}  
  
@Composable  
fun ScreenRateUs(navController: NavHostController) {  
 // *TODO: Implement ScreenRateUs content*}  
  
@Composable  
fun ScreenShare(navController: NavHostController) {  
 // *TODO: Implement ScreenShare content*}  
  
@Composable  
fun ScreenLogin(navController: NavHostController) {  
 // *TODO: Implement ScreenLogin content*}  
  
@Composable  
fun ScreenSettings(navController: NavHostController) {  
 // *TODO: Implement ScreenSettings content*}  
  
@Composable  
fun ScreenTrash(navController: NavHostController) {  
 // *TODO: Implement ScreenTrash content*}  
  
@Composable  
fun ScreenSync(navController: NavHostController) {  
 // *TODO: Implement ScreenSync content*}  
  
@Composable  
fun ScreenMessage(navController: NavHostController) {  
 // *TODO: Implement ScreenMessage content*}  
  
@OptIn(ExperimentalMaterial3Api::class)  
@Composable  
fun Screen1(navController: NavHostController) {  
 Scaffold(  
 topBar = **{** CenterAlignedTopAppBar(  
 title = **{** Text("Screen 1")  
 **}**,  
 navigationIcon = **{** IconButton(onClick = **{** navController.navigate("main") **}**) **{** Icon(  
 imageVector = Icons.Filled.*ArrowBack*,  
 contentDescription = "Back"  
 )  
 **}  
 }** )  
 **}** ) **{** Column(  
 modifier = Modifier  
 .*fillMaxSize*()  
 .*padding*(**it**),  
 horizontalAlignment = Alignment.CenterHorizontally,  
 verticalArrangement = Arrangement.Center  
 ) **{** Text(text = "This is Screen 1")  
 **}  
 }**}  
  
@OptIn(ExperimentalMaterial3Api::class)  
@Composable  
fun Screen2(navController: NavHostController) {  
 Scaffold(  
 topBar = **{** CenterAlignedTopAppBar(  
 title = **{** Text("Screen 2")  
 **}**,  
 navigationIcon = **{** IconButton(onClick = **{** navController.navigate("main") **}**) **{** Icon(  
 imageVector = Icons.Filled.*ArrowBack*,  
 contentDescription = "Back"  
 )  
 **}  
 }** )  
 **}** ) **{** Column(  
 modifier = Modifier  
 .*fillMaxSize*()  
 .*padding*(**it**),  
 horizontalAlignment = Alignment.CenterHorizontally,  
 verticalArrangement = Arrangement.Center  
 ) **{** Text(text = "This is Screen 2")  
 **}  
 }**}

**Sample 5**

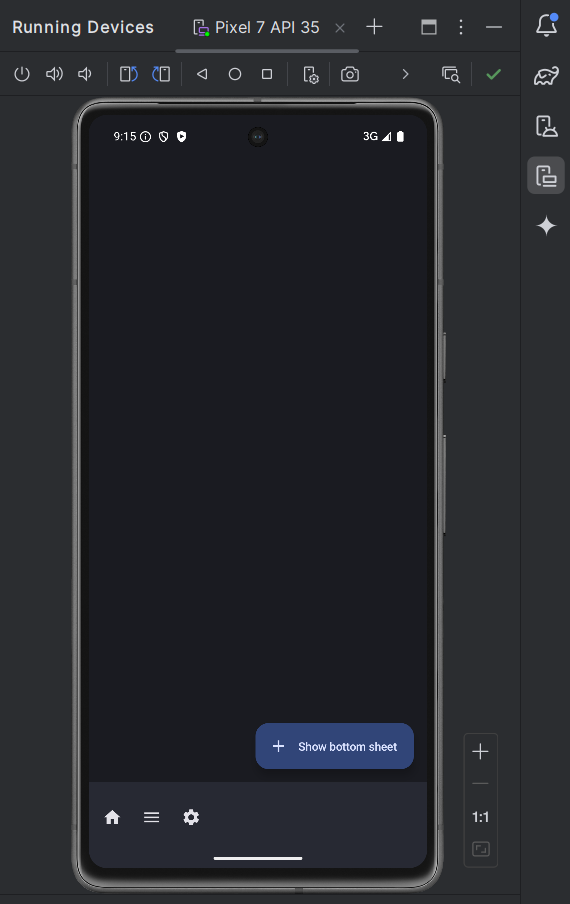


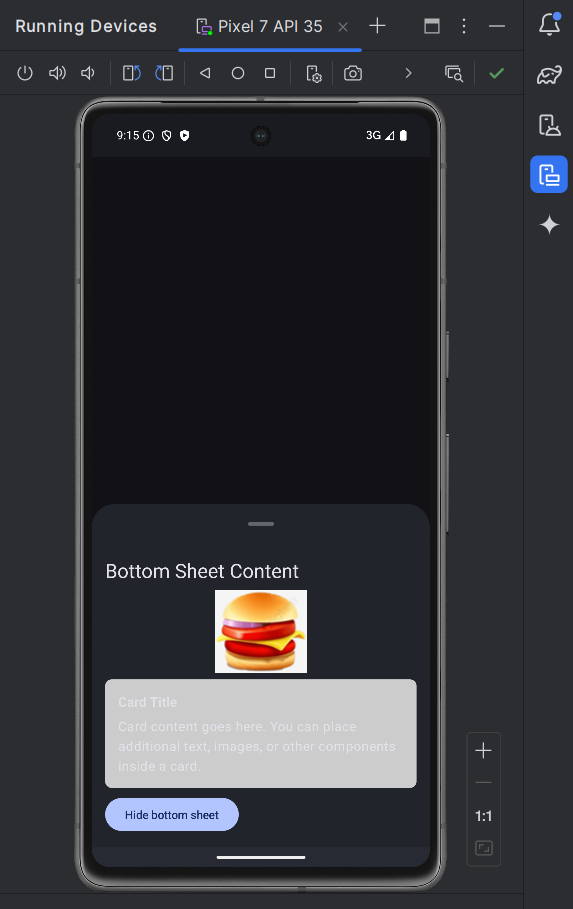




package com.example.appbarcenteraligned  
  
import android.os.Bundle  
import androidx.activity.ComponentActivity  
import androidx.activity.compose.setContent  
import androidx.activity.enableEdgeToEdge  
import androidx.compose.foundation.Image  
import androidx.compose.foundation.background  
import androidx.compose.foundation.clickable  
import androidx.compose.foundation.layout.\*  
import androidx.compose.foundation.rememberScrollState  
import androidx.compose.foundation.shape.*CircleShape*import androidx.compose.foundation.verticalScroll  
import androidx.compose.material.icons.Icons  
import androidx.compose.material.icons.filled.\*  
import androidx.compose.material3.\*  
import androidx.compose.runtime.\*  
import androidx.compose.ui.Alignment  
import androidx.compose.ui.Modifier  
import androidx.compose.ui.draw.clip  
import androidx.compose.ui.graphics.Color  
import androidx.compose.ui.graphics.painter.Painter  
import androidx.compose.ui.graphics.vector.ImageVector  
import androidx.compose.ui.input.nestedscroll.nestedScroll  
import androidx.compose.ui.res.painterResource  
import androidx.compose.ui.text.font.FontWeight  
import androidx.compose.ui.text.style.TextOverflow  
import androidx.compose.ui.unit.dp  
import androidx.navigation.NavHostController  
import androidx.navigation.compose.NavHost  
import androidx.navigation.compose.composable  
import androidx.navigation.compose.rememberNavController  
import com.example.appbarcenteraligned.ui.theme.AppBarCenterAlignedTheme  
import kotlinx.coroutines.CoroutineScope  
import kotlinx.coroutines.launch  
  
class MainActivity : ComponentActivity() {  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 *enableEdgeToEdge*()  
 *setContent* **{** AppBarCenterAlignedTheme **{** val navController = rememberNavController()  
 NavigationComponent(navController)  
 **}  
 }** }  
}  
  
@OptIn(ExperimentalMaterial3Api::class)  
@Composable  
fun CenterAlignedTopAppBarExample(navController: NavHostController) {  
 val drawerState = rememberDrawerState(DrawerValue.*Closed*)  
 val coroutineScope = rememberCoroutineScope()  
 val scrollBehavior = TopAppBarDefaults.pinnedScrollBehavior(rememberTopAppBarState())  
 val showDialog = remember **{** *mutableStateOf*(false) **}** val snackbarHostState = remember **{** SnackbarHostState() **}** val scope = rememberCoroutineScope()  
 val imagePainter = painterResource(id = R.drawable.*\_d\_burger*) // Replace with your image resource  
  
 if (showDialog.value) {  
 AdvancedAlertDialog(  
 showDialog = showDialog,  
 onNavigateToScreen1 = **{** navController.navigate("screen1") **}**,  
 onNavigateToScreen2 = **{** navController.navigate("screen2") **}**,  
 imagePainter = imagePainter  
 )  
 }  
  
 ModalNavigationDrawer(  
 drawerState = drawerState,  
 drawerContent = **{** DrawerContent(navController, drawerState, coroutineScope)  
 **}** ) **{** Scaffold(  
 modifier = Modifier.*nestedScroll*(scrollBehavior.nestedScrollConnection),  
 topBar = **{** CenterAlignedTopAppBar(  
 colors = TopAppBarDefaults.topAppBarColors(  
 containerColor = MaterialTheme.colorScheme.primaryContainer,  
 titleContentColor = MaterialTheme.colorScheme.primary,  
 ),  
 title = **{** Text(  
 "Centered Top App Bar",  
 maxLines = 1,  
 overflow = TextOverflow.Ellipsis  
 )  
 **}**,  
 navigationIcon = **{** IconButton(onClick = **{** coroutineScope.*launch* **{** drawerState.open() **} }**) **{** Icon(  
 imageVector = Icons.Filled.*Menu*,  
 contentDescription = "Menu"  
 )  
 **}  
 }**,  
 actions = **{** IconButton(onClick = **{** navController.popBackStack() **}**) **{** Icon(  
 imageVector = Icons.Filled.*ArrowBack*,  
 contentDescription = "Back"  
 )  
 **}  
 }**,  
 scrollBehavior = scrollBehavior,  
 )  
 **}**,  
 bottomBar = **{** BottomAppBar(  
 actions = **{** IconButton(onClick = **{** scope.*launch* **{** val result = snackbarHostState.showSnackbar(  
 message = "Luis this is a Snackbar",  
 actionLabel = "Action",  
 duration = SnackbarDuration.*Indefinite* )  
 when (result) {  
 SnackbarResult.*ActionPerformed* -> {  
 navController.navigate("screen1")  
 }  
 SnackbarResult.*Dismissed* -> {  
 // Handle snackbar dismissed  
 }  
 }  
 **}  
 }**) **{** Icon(imageVector = Icons.Filled.*Home*, contentDescription = "Button 1")  
 **}** IconButton(onClick = **{** /\* Handle bottom button 2 click \*/ **}**) **{** Icon(imageVector = Icons.Filled.*AccountCircle*, contentDescription = "Button 2")  
 **}** IconButton(onClick = **{** /\* Handle bottom button 3 click \*/ **}**) **{** Icon(imageVector = Icons.Filled.*Call*, contentDescription = "Button 3")  
 **}  
 }**,  
 floatingActionButton = **{** FloatingActionButton(onClick = **{** showDialog.value = true **}**) **{** Icon(imageVector = Icons.Filled.*Add*, contentDescription = "Floating Button")  
 **}  
 }** )  
 **}**,  
 floatingActionButtonPosition = FabPosition.End,  
 snackbarHost = **{** SnackbarHost(hostState = snackbarHostState)  
 **}** ) **{** innerPadding **->** MyScrollContent(innerPadding)  
 **}  
 }**}  
  
@Composable  
fun AdvancedAlertDialog(  
 showDialog: MutableState<Boolean>,  
 onNavigateToScreen1: () -> Unit,  
 onNavigateToScreen2: () -> Unit,  
 imagePainter: Painter  
) {  
 AlertDialog(  
 onDismissRequest = **{** showDialog.value = false **}**,  
 title = **{** Text("Floating Button Pressed") **}**,  
 text = **{** Column **{** Image(  
 painter = imagePainter,  
 contentDescription = null,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*height*(150.*dp*)  
 .*padding*(bottom = 16.*dp*)  
 )  
 Text("You pressed on the Floating button")  
 **}  
 }**,  
 confirmButton = **{** Row(  
 modifier = Modifier.*fillMaxWidth*(),  
 horizontalArrangement = Arrangement.SpaceBetween  
 ) **{** TextButton(onClick = **{** showDialog.value = false  
 onNavigateToScreen1()  
 **}**) **{** Text("Screen 1")  
 **}** TextButton(onClick = **{** showDialog.value = false  
 onNavigateToScreen2()  
 **}**) **{** Text("Screen 2")  
 **}  
 }  
 }** )  
}  
  
@Composable  
fun DrawerContent(navController: NavHostController, drawerState: DrawerState, coroutineScope: CoroutineScope) {  
 ModalDrawerSheet **{** Text("Drawer title", modifier = Modifier.*padding*(16.*dp*))  
 Divider()  
 DrawerItem(icon = Icons.Filled.*Home*, label = "Screen 1", onClick = **{** coroutineScope.*launch* **{** drawerState.close() **}** navController.navigate("screen1")  
 **}**)  
 DrawerItem(icon = Icons.Filled.*MailOutline*, label = "Screen 2", onClick = **{** coroutineScope.*launch* **{** drawerState.close() **}** navController.navigate("screen2")  
 **}**)  
 DrawerItem(icon = Icons.Filled.*Star*, label = "Sync", onClick = **{** coroutineScope.*launch* **{** drawerState.close() **}** navController.navigate("sync")  
 **}**)  
 DrawerItem(icon = Icons.Filled.*Delete*, label = "Trash", onClick = **{** coroutineScope.*launch* **{** drawerState.close() **}** navController.navigate("trash")  
 **}**)  
 DrawerItem(icon = Icons.Filled.*Settings*, label = "Settings", onClick = **{** coroutineScope.*launch* **{** drawerState.close() **}** navController.navigate("settings")  
 **}**)  
  
 // Profile and Communicate Sections  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
 Text("Profile", modifier = Modifier.*padding*(start = 16.*dp*))  
 DrawerItem(icon = Icons.Filled.*AccountBox*, label = "Login", onClick = **{** coroutineScope.*launch* **{** drawerState.close() **}** navController.navigate("login")  
 **}**)  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
 Text("Communicate", modifier = Modifier.*padding*(start = 16.*dp*))  
 DrawerItem(icon = Icons.Filled.*Share*, label = "Share", onClick = **{** coroutineScope.*launch* **{** drawerState.close() **}** navController.navigate("share")  
 **}**)  
 DrawerItem(icon = Icons.Filled.*ThumbUp*, label = "Rate us", onClick = **{** coroutineScope.*launch* **{** drawerState.close() **}** navController.navigate("rate\_us")  
 **}**)  
 // ...other drawer items  
 **}**}  
  
@Composable  
fun DrawerItem(icon: ImageVector, label: String, onClick: () -> Unit) {  
 Row(  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*clickable*(onClick = onClick)  
 .*padding*(16.*dp*),  
 verticalAlignment = Alignment.CenterVertically  
 ) **{** Icon(icon, contentDescription = label)  
 Spacer(modifier = Modifier.*width*(16.*dp*))  
 Text(label)  
 **}**}  
  
@Composable  
fun MyScrollContent(padding: PaddingValues) {  
 val scrollState = rememberScrollState()  
 Column(  
 modifier = Modifier  
 .*padding*(padding)  
 .*verticalScroll*(scrollState)  
 .*fillMaxSize*()  
 ) **{** *repeat*(20) **{** index **->** Text(  
 text = "Item $index",  
 modifier = Modifier  
 .*padding*(16.*dp*)  
 .*fillMaxWidth*()  
 )  
 **}  
 }**}  
  
@Composable  
fun NavigationComponent(navController: NavHostController) {  
 NavHost(navController, startDestination = "main") **{** *composable*("main") **{** CenterAlignedTopAppBarExample(navController) **}** *composable*("screen1") **{** Screen1(navController) **}** *composable*("screen2") **{** Screen2(navController) **}** *composable*("message") **{** ScreenMessage(navController) **}** *composable*("sync") **{** ScreenSync(navController) **}** *composable*("trash") **{** ScreenTrash(navController) **}** *composable*("settings") **{** ScreenSettings(navController) **}** *composable*("login") **{** ScreenLogin(navController) **}** *composable*("share") **{** ScreenShare(navController) **}** *composable*("rate\_us") **{** ScreenRateUs(navController) **}  
 }**}  
  
@Composable  
fun ScreenRateUs(navController: NavHostController) {  
 // *TODO: Implement ScreenRateUs content*}  
  
@Composable  
fun ScreenShare(navController: NavHostController) {  
 // *TODO: Implement ScreenShare content*}  
  
@Composable  
fun ScreenLogin(navController: NavHostController) {  
 // *TODO: Implement ScreenLogin content*}  
  
@Composable  
fun ScreenSettings(navController: NavHostController) {  
 // *TODO: Implement ScreenSettings content*}  
  
@Composable  
fun ScreenTrash(navController: NavHostController) {  
 // *TODO: Implement ScreenTrash content*}  
  
@Composable  
fun ScreenSync(navController: NavHostController) {  
 // *TODO: Implement ScreenSync content*}  
  
@Composable  
fun ScreenMessage(navController: NavHostController) {  
 // *TODO: Implement ScreenMessage content*}  
  
@OptIn(ExperimentalMaterial3Api::class)  
@Composable  
fun Screen1(navController: NavHostController) {  
 Scaffold(  
 topBar = **{** CenterAlignedTopAppBar(  
 title = **{** Text("Screen 1")  
 **}**,  
 navigationIcon = **{** IconButton(onClick = **{** navController.navigate("main") **}**) **{** Icon(  
 imageVector = Icons.Filled.*ArrowBack*,  
 contentDescription = "Back"  
 )  
 **}  
 }** )  
 **}** ) **{** Column(  
 modifier = Modifier  
 .*fillMaxSize*()  
 .*padding*(**it**),  
 horizontalAlignment = Alignment.CenterHorizontally,  
 verticalArrangement = Arrangement.Center  
 ) **{** Text(text = "This is Screen 1")  
 **}  
 }**}  
  
@OptIn(ExperimentalMaterial3Api::class)  
@Composable  
fun Screen2(navController: NavHostController) {  
 Scaffold(  
 topBar = **{** CenterAlignedTopAppBar(  
 title = **{** Text("Screen 2")  
 **}**,  
 navigationIcon = **{** IconButton(onClick = **{** navController.navigate("main") **}**) **{** Icon(  
 imageVector = Icons.Filled.*ArrowBack*,  
 contentDescription = "Back"  
 )  
 **}  
 }** )  
 **}** ) **{** Column(  
 modifier = Modifier  
 .*fillMaxSize*()  
 .*padding*(**it**),  
 horizontalAlignment = Alignment.CenterHorizontally,  
 verticalArrangement = Arrangement.Center  
 ) **{** Text(text = "This is Screen 2")  
 **}  
 }**}  
  
// Add more screen composables as needed for message, sync, trash, settings, login, share, and rate\_us

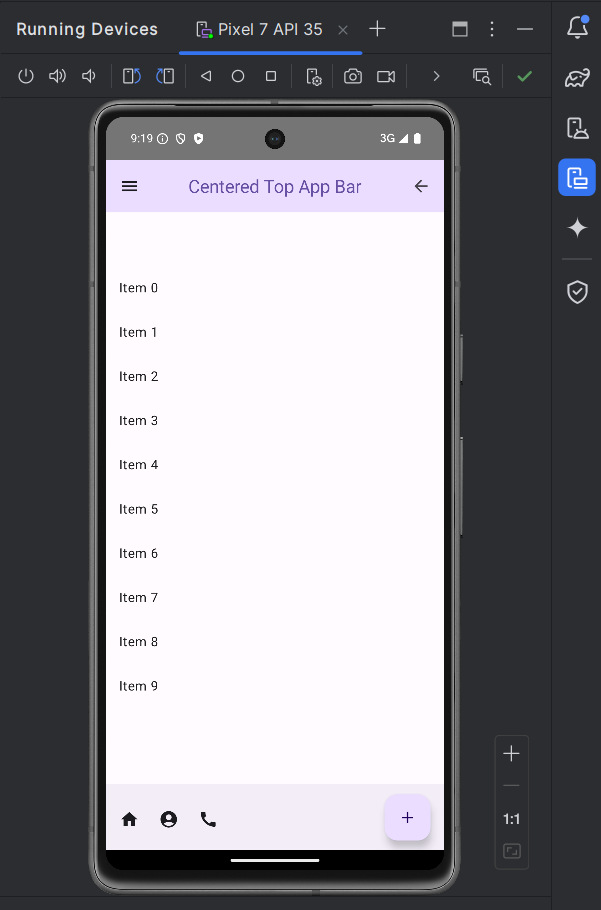
**Sample 6**

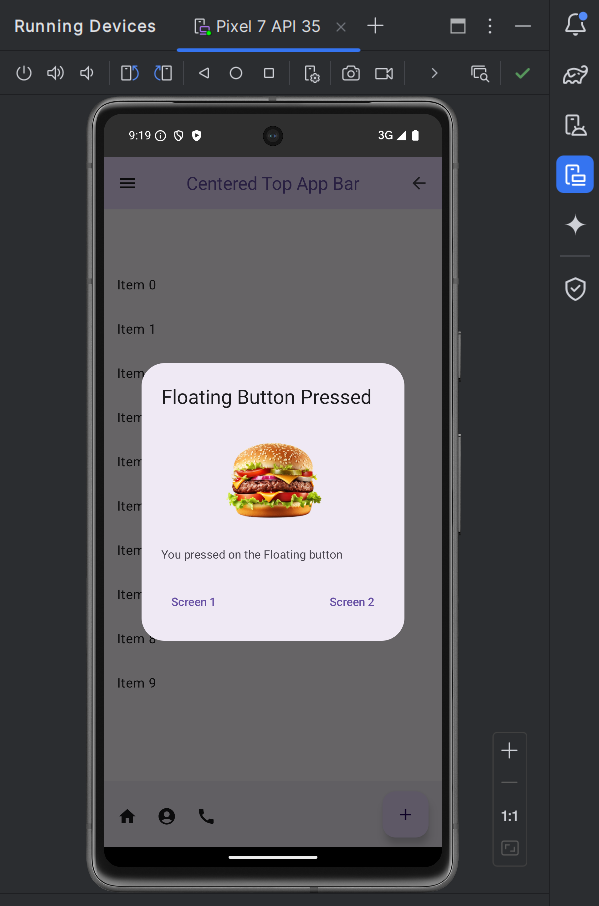


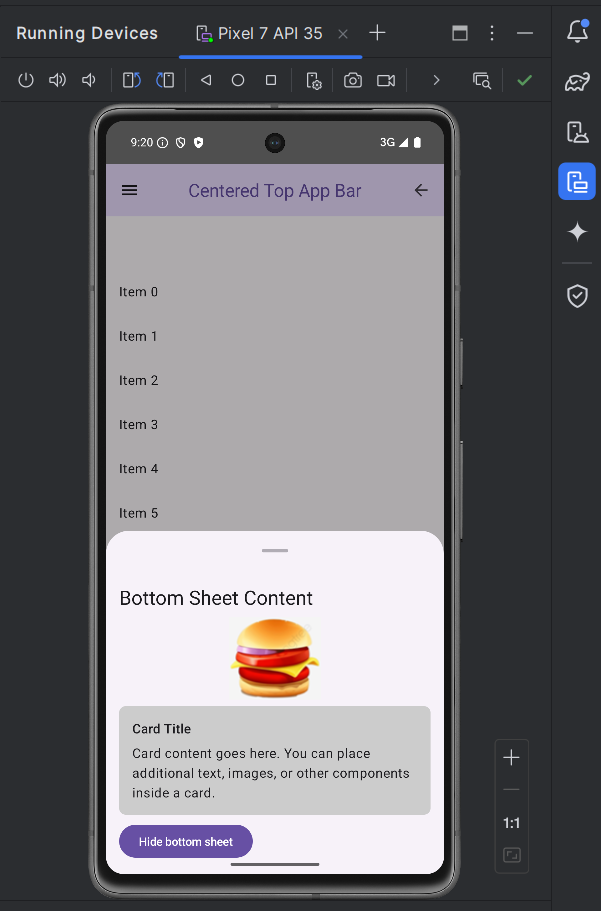


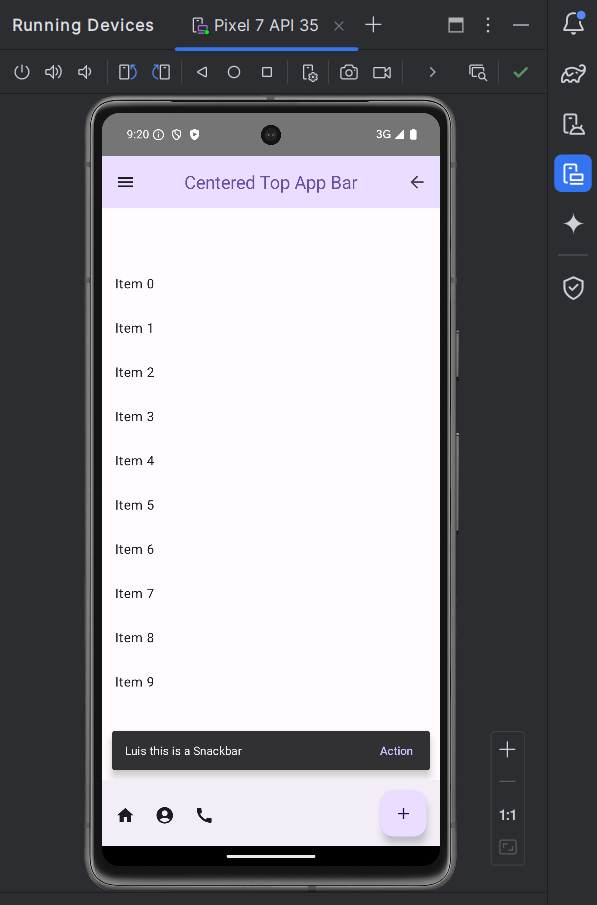
package com.example.bottomsheet  
  
import android.os.Bundle  
import androidx.activity.ComponentActivity  
import androidx.activity.compose.setContent  
import androidx.activity.enableEdgeToEdge  
import androidx.compose.foundation.Image  
import androidx.compose.foundation.layout.\*  
import androidx.compose.foundation.shape.RoundedCornerShape  
import androidx.compose.material.icons.Icons  
import androidx.compose.material.icons.filled.*Add*import androidx.compose.material.icons.filled.*Home*import androidx.compose.material.icons.filled.*Menu*import androidx.compose.material.icons.filled.*Settings*import androidx.compose.material3.\*  
import androidx.compose.runtime.\*  
import androidx.compose.ui.Modifier  
import androidx.compose.ui.draw.clip  
import androidx.compose.ui.graphics.Color  
import androidx.compose.ui.layout.ContentScale  
import androidx.compose.ui.res.painterResource  
import androidx.compose.ui.tooling.preview.Preview  
import androidx.compose.ui.unit.dp  
import com.example.bottomsheet.ui.theme.BottomSheetTheme  
import kotlinx.coroutines.launch  
  
class MainActivity : ComponentActivity() {  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 *enableEdgeToEdge*()  
 *setContent* **{** BottomSheetTheme **{** MainScreen()  
 **}  
 }** }  
}  
  
@OptIn(ExperimentalMaterial3Api::class)  
@Composable  
fun MainScreen() {  
 val sheetState = rememberModalBottomSheetState()  
 val scope = rememberCoroutineScope()  
 var showBottomSheet by remember **{** *mutableStateOf*(false) **}** Scaffold(  
 modifier = Modifier.*fillMaxSize*(),  
 bottomBar = **{** BottomAppBar **{** IconButton(onClick = **{** showBottomSheet = true **}**) **{** Icon(Icons.Filled.*Home*, contentDescription = "Home")  
 **}** IconButton(onClick = **{** /\* Do something \*/ **}**) **{** Icon(Icons.Filled.*Menu*, contentDescription = "Menu")  
 **}** IconButton(onClick = **{** /\* Do something \*/ **}**) **{** Icon(Icons.Filled.*Settings*, contentDescription = "Settings")  
 **}  
 }  
 }**,  
 floatingActionButton = **{** ExtendedFloatingActionButton(  
 text = **{** Text("Show bottom sheet") **}**,  
 icon = **{** Icon(Icons.Filled.*Add*, contentDescription = null) **}**,  
 onClick = **{** showBottomSheet = true  
 **}** )  
 **}** ) **{** innerPadding **->** // Apply the innerPadding to the content  
 Box(modifier = Modifier.*padding*(innerPadding).*fillMaxSize*()) **{** if (showBottomSheet) {  
 ModalBottomSheet(  
 onDismissRequest = **{** showBottomSheet = false  
 **}**,  
 sheetState = sheetState  
 ) **{** // Sheet content  
 Column(  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*padding*(16.*dp*)  
 ) **{** Text("Bottom Sheet Content", style = MaterialTheme.typography.headlineSmall)  
 Spacer(modifier = Modifier.*height*(8.*dp*))  
 Image(  
 painter = painterResource(id = R.drawable.*burger2*), // Replace with your image resource  
 contentDescription = "Example Image",  
 contentScale = ContentScale.Fit,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*height*(100.*dp*)  
 .*clip*(*RoundedCornerShape*(8.*dp*)) // Optional: Adds rounded corners to the image  
 )  
 Spacer(modifier = Modifier.*height*(8.*dp*))  
 Card(  
 shape = *RoundedCornerShape*(8.*dp*),  
 modifier = Modifier.*fillMaxWidth*(),  
 colors = CardDefaults.cardColors(containerColor = Color.LightGray)  
 ) **{** Column(modifier = Modifier.*padding*(16.*dp*)) **{** Text("Card Title", style = MaterialTheme.typography.titleMedium)  
 Spacer(modifier = Modifier.*height*(8.*dp*))  
 Text("Card content goes here. You can place additional text, images, or other components inside a card.")  
 **}  
 }** Spacer(modifier = Modifier.*height*(8.*dp*))  
 Button(onClick = **{** scope.*launch* **{** sheetState.hide() **}**.invokeOnCompletion **{** if (!sheetState.isVisible) {  
 showBottomSheet = false  
 }  
 **}  
 }**) **{** Text("Hide bottom sheet")  
 **}  
 }  
 }** }  
 **}  
 }**}  
  
@Preview(showBackground = true)  
@Composable  
fun MainScreenPreview() {  
 BottomSheetTheme **{** MainScreen()  
 **}**}

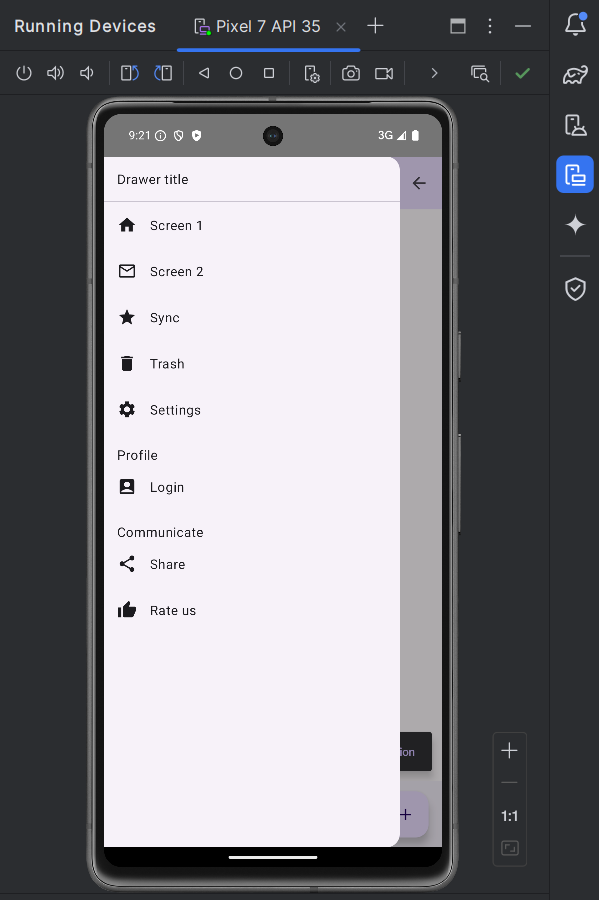
**Sample 7**





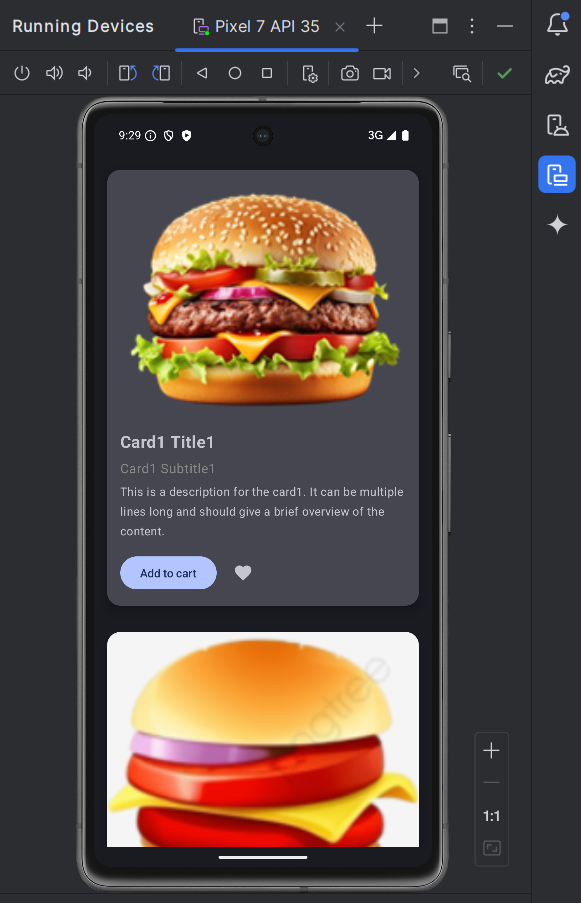


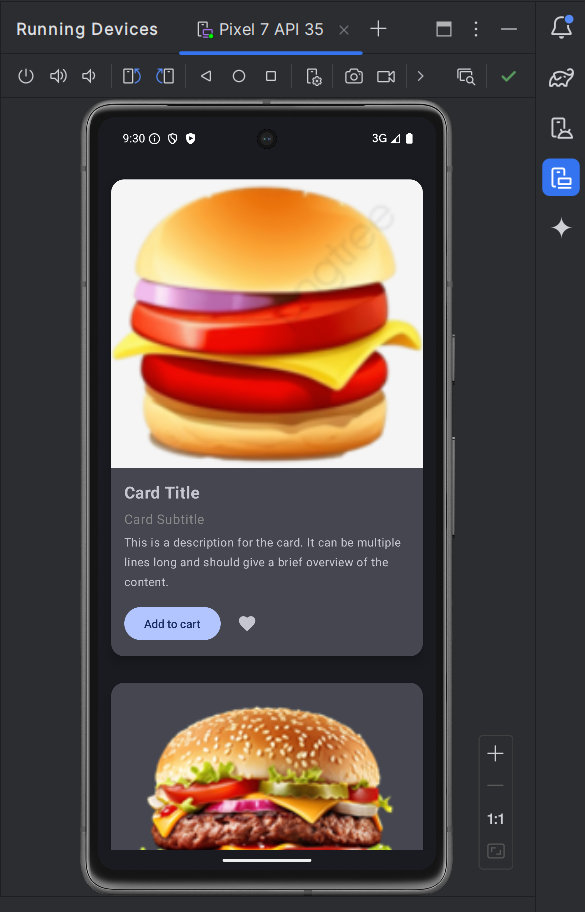


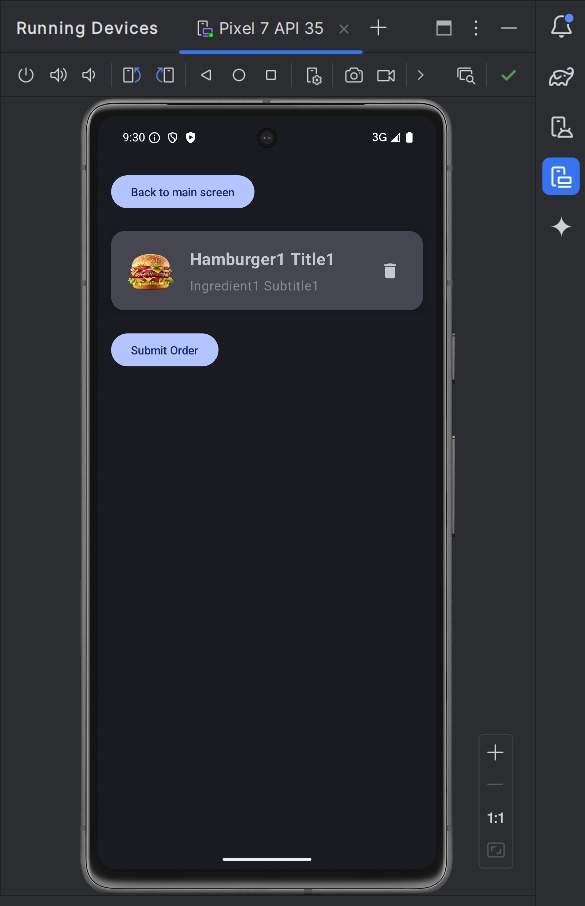


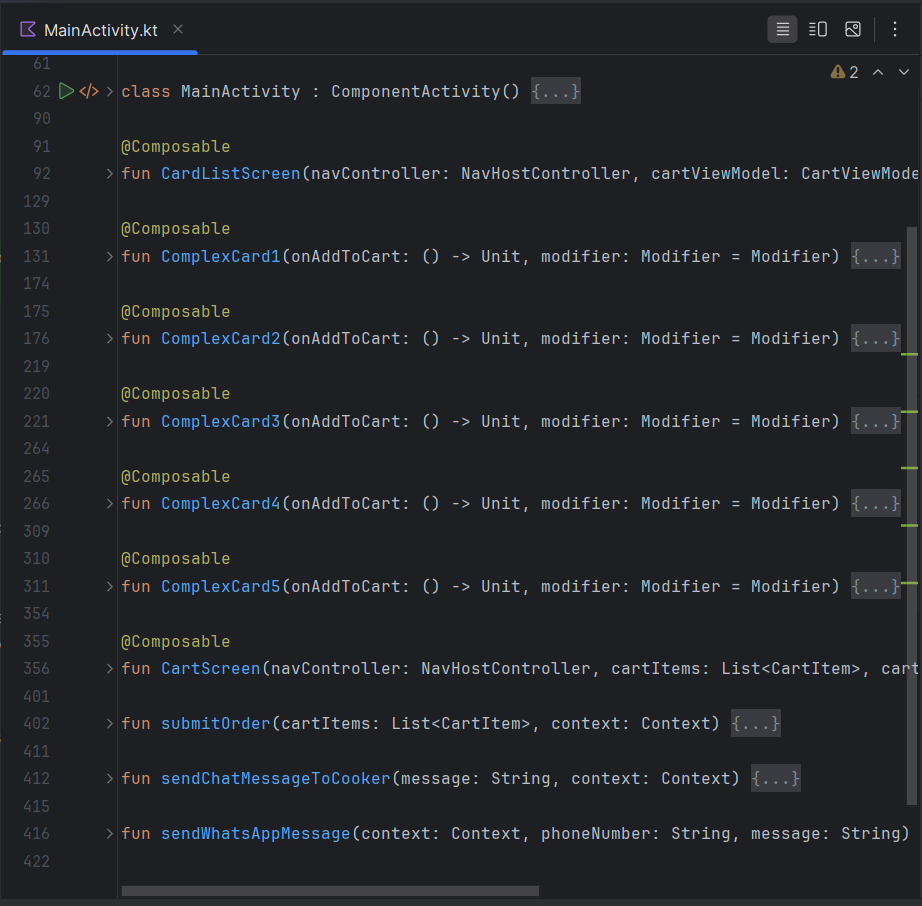
package com.example.appbarcenteraligned  
  
import android.os.Bundle  
import androidx.activity.ComponentActivity  
import androidx.activity.compose.setContent  
import androidx.compose.foundation.Image  
import androidx.compose.foundation.clickable  
import androidx.compose.foundation.layout.\*  
import androidx.compose.foundation.rememberScrollState  
import androidx.compose.foundation.shape.RoundedCornerShape  
import androidx.compose.foundation.verticalScroll  
import androidx.compose.material.icons.Icons  
import androidx.compose.material.icons.filled.\*  
import androidx.compose.material3.\*  
import androidx.compose.runtime.\*  
import androidx.compose.ui.Alignment  
import androidx.compose.ui.Modifier  
import androidx.compose.ui.draw.clip  
import androidx.compose.ui.graphics.Color  
import androidx.compose.ui.graphics.painter.Painter  
import androidx.compose.ui.graphics.vector.ImageVector  
import androidx.compose.ui.input.nestedscroll.nestedScroll  
import androidx.compose.ui.layout.ContentScale  
import androidx.compose.ui.res.painterResource  
import androidx.compose.ui.text.style.TextOverflow  
import androidx.compose.ui.unit.dp  
import androidx.navigation.NavHostController  
import androidx.navigation.compose.NavHost  
import androidx.navigation.compose.composable  
import androidx.navigation.compose.rememberNavController  
import kotlinx.coroutines.CoroutineScope  
import kotlinx.coroutines.launch  
  
class MainActivity : ComponentActivity() {  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 *setContent* **{** AppBarCenterAlignedTheme **{** val navController = rememberNavController()  
 NavigationComponent(navController)  
 **}  
 }** }  
}  
  
@Composable  
fun NavigationComponent(navController: NavHostController) {  
 NavHost(navController, startDestination = "main") **{** *composable*("main") **{** CenterAlignedTopAppBarExample(navController) **}** *composable*("screen1") **{** Screen1(navController) **}** *composable*("screen2") **{** Screen2(navController) **}** *composable*("message") **{** ScreenMessage(navController) **}** *composable*("sync") **{** ScreenSync(navController) **}** *composable*("trash") **{** ScreenTrash(navController) **}** *composable*("settings") **{** ScreenSettings(navController) **}** *composable*("login") **{** ScreenLogin(navController) **}** *composable*("share") **{** ScreenShare(navController) **}** *composable*("rate\_us") **{** ScreenRateUs(navController) **}  
 }**}  
  
@OptIn(ExperimentalMaterial3Api::class)  
@Composable  
fun CenterAlignedTopAppBarExample(navController: NavHostController) {  
 val drawerState = rememberDrawerState(DrawerValue.*Closed*)  
 val coroutineScope = rememberCoroutineScope()  
 val scrollBehavior = TopAppBarDefaults.pinnedScrollBehavior(rememberTopAppBarState())  
 val showDialog = remember **{** *mutableStateOf*(false) **}** val snackbarHostState = remember **{** SnackbarHostState() **}** val scope = rememberCoroutineScope()  
 val imagePainter = painterResource(id = R.drawable.*\_d\_burger*) // Replace with your image resource  
  
 if (showDialog.value) {  
 AdvancedAlertDialog(  
 showDialog = showDialog,  
 onNavigateToScreen1 = **{** navController.navigate("screen1") **}**,  
 onNavigateToScreen2 = **{** navController.navigate("screen2") **}**,  
 imagePainter = imagePainter  
 )  
 }  
  
 var showBottomSheet by remember **{** *mutableStateOf*(false) **}** val sheetState = rememberModalBottomSheetState()  
  
 ModalNavigationDrawer(  
 drawerState = drawerState,  
 drawerContent = **{** DrawerContent(navController, drawerState, coroutineScope)  
 **}** ) **{** Scaffold(  
 modifier = Modifier.*nestedScroll*(scrollBehavior.nestedScrollConnection),  
 topBar = **{** CenterAlignedTopAppBar(  
 colors = TopAppBarDefaults.topAppBarColors(  
 containerColor = MaterialTheme.colorScheme.primaryContainer,  
 titleContentColor = MaterialTheme.colorScheme.primary,  
 ),  
 title = **{** Text(  
 "Centered Top App Bar",  
 maxLines = 1,  
 overflow = TextOverflow.Ellipsis  
 )  
 **}**,  
 navigationIcon = **{** IconButton(onClick = **{** coroutineScope.*launch* **{** drawerState.open() **} }**) **{** Icon(  
 imageVector = Icons.Filled.*Menu*,  
 contentDescription = "Menu"  
 )  
 **}  
 }**,  
 actions = **{** IconButton(onClick = **{** navController.popBackStack() **}**) **{** Icon(  
 imageVector = Icons.Filled.*ArrowBack*,  
 contentDescription = "Back"  
 )  
 **}  
 }**,  
 scrollBehavior = scrollBehavior,  
 )  
 **}**,  
 bottomBar = **{** BottomAppBar(  
 actions = **{** IconButton(onClick = **{** scope.*launch* **{** val result = snackbarHostState.showSnackbar(  
 message = "Luis this is a Snackbar",  
 actionLabel = "Action",  
 duration = SnackbarDuration.*Indefinite* )  
 when (result) {  
 SnackbarResult.*ActionPerformed* -> {  
 navController.navigate("screen1")  
 }  
 SnackbarResult.*Dismissed* -> {  
 // Handle snackbar dismissed  
 }  
 }  
 **}  
 }**) **{** Icon(imageVector = Icons.Filled.*Home*, contentDescription = "Button 1")  
 **}** IconButton(onClick = **{** showBottomSheet = true **}**) **{** Icon(imageVector = Icons.Filled.*AccountCircle*, contentDescription = "Button 2")  
 **}** IconButton(onClick = **{** /\* Handle bottom button 3 click \*/ **}**) **{** Icon(imageVector = Icons.Filled.*Call*, contentDescription = "Button 3")  
 **}  
 }**,  
 floatingActionButton = **{** FloatingActionButton(onClick = **{** showDialog.value = true **}**) **{** Icon(imageVector = Icons.Filled.*Add*, contentDescription = "Floating Button")  
 **}  
 }** )  
 **}**,  
 floatingActionButtonPosition = FabPosition.End,  
 snackbarHost = **{** SnackbarHost(hostState = snackbarHostState)  
 **}** ) **{** innerPadding **->** Box(modifier = Modifier.*padding*(innerPadding).*fillMaxSize*()) **{** if (showBottomSheet) {  
 ModalBottomSheet(  
 onDismissRequest = **{** showBottomSheet = false  
 **}**,  
 sheetState = sheetState  
 ) **{** BottomSheetContent **{** scope.*launch* **{** sheetState.hide() **}**.invokeOnCompletion **{** if (!sheetState.isVisible) {  
 showBottomSheet = false  
 }  
 **}  
 }  
 }** }  
 MyScrollContent(innerPadding)  
 **}  
 }  
 }**}  
  
@Composable  
fun BottomSheetContent(onHide: () -> Unit) {  
 Column(  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*padding*(16.*dp*)  
 ) **{** Text("Bottom Sheet Content", style = MaterialTheme.typography.headlineSmall)  
 Spacer(modifier = Modifier.*height*(8.*dp*))  
 Image(  
 painter = painterResource(id = R.drawable.*burger2*), // Replace with your image resource  
 contentDescription = "Example Image",  
 contentScale = ContentScale.Fit,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*height*(100.*dp*)  
 .*clip*(*RoundedCornerShape*(8.*dp*)) // Optional: Adds rounded corners to the image  
 )  
 Spacer(modifier = Modifier.*height*(8.*dp*))  
 Card(  
 shape = *RoundedCornerShape*(8.*dp*),  
 modifier = Modifier.*fillMaxWidth*(),  
 colors = CardDefaults.cardColors(containerColor = Color.LightGray)  
 ) **{** Column(modifier = Modifier.*padding*(16.*dp*)) **{** Text("Card Title", style = MaterialTheme.typography.titleMedium)  
 Spacer(modifier = Modifier.*height*(8.*dp*))  
 Text("Card content goes here. You can place additional text, images, or other components inside a card.")  
 **}  
 }** Spacer(modifier = Modifier.*height*(8.*dp*))  
 Button(onClick = onHide) **{** Text("Hide bottom sheet")  
 **}  
 }**}  
  
@Composable  
fun AdvancedAlertDialog(  
 showDialog: MutableState<Boolean>,  
 onNavigateToScreen1: () -> Unit,  
 onNavigateToScreen2: () -> Unit,  
 imagePainter: Painter  
) {  
 AlertDialog(  
 onDismissRequest = **{** showDialog.value = false **}**,  
 title = **{** Text("Floating Button Pressed") **}**,  
 text = **{** Column **{** Image(  
 painter = imagePainter,  
 contentDescription = null,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*height*(150.*dp*)  
 .*padding*(bottom = 16.*dp*)  
 )  
 Text("You pressed on the Floating button")  
 **}  
 }**,  
 confirmButton = **{** Row(  
 modifier = Modifier.*fillMaxWidth*(),  
 horizontalArrangement = Arrangement.SpaceBetween  
 ) **{** TextButton(onClick = **{** showDialog.value = false  
 onNavigateToScreen1()  
 **}**) **{** Text("Screen 1")  
 **}** TextButton(onClick = **{** showDialog.value = false  
 onNavigateToScreen2()  
 **}**) **{** Text("Screen 2")  
 **}  
 }  
 }** )  
}  
  
@Composable  
fun DrawerContent(navController: NavHostController, drawerState: DrawerState, coroutineScope: CoroutineScope) {  
 ModalDrawerSheet **{** Text("Drawer title", modifier = Modifier.*padding*(16.*dp*))  
 Divider()  
 DrawerItem(icon = Icons.Filled.*Home*, label = "Screen 1", onClick = **{** coroutineScope.*launch* **{** drawerState.close() **}** navController.navigate("screen1")  
 **}**)  
 DrawerItem(icon = Icons.Filled.*MailOutline*, label = "Screen 2", onClick = **{** coroutineScope.*launch* **{** drawerState.close() **}** navController.navigate("screen2")  
 **}**)  
 DrawerItem(icon = Icons.Filled.*Star*, label = "Sync", onClick = **{** coroutineScope.*launch* **{** drawerState.close() **}** navController.navigate("sync")  
 **}**)  
 DrawerItem(icon = Icons.Filled.*Delete*, label = "Trash", onClick = **{** coroutineScope.*launch* **{** drawerState.close() **}** navController.navigate("trash")  
 **}**)  
 DrawerItem(icon = Icons.Filled.*Settings*, label = "Settings", onClick = **{** coroutineScope.*launch* **{** drawerState.close() **}** navController.navigate("settings")  
 **}**)  
  
 // Profile and Communicate Sections  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
 Text("Profile", modifier = Modifier.*padding*(start = 16.*dp*))  
 DrawerItem(icon = Icons.Filled.*AccountBox*, label = "Login", onClick = **{** coroutineScope.*launch* **{** drawerState.close() **}** navController.navigate("login")  
 **}**)  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
 Text("Communicate", modifier = Modifier.*padding*(start = 16.*dp*))  
 DrawerItem(icon = Icons.Filled.*Share*, label = "Share", onClick = **{** coroutineScope.*launch* **{** drawerState.close() **}** navController.navigate("share")  
 **}**)  
 DrawerItem(icon = Icons.Filled.*ThumbUp*, label = "Rate us", onClick = **{** coroutineScope.*launch* **{** drawerState.close() **}** navController.navigate("rate\_us")  
 **}**)  
 // ...other drawer items  
 **}**}  
  
@Composable  
fun DrawerItem(icon: ImageVector, label: String, onClick: () -> Unit) {  
 Row(  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*clickable*(onClick = onClick)  
 .*padding*(16.*dp*),  
 verticalAlignment = Alignment.CenterVertically  
 ) **{** Icon(icon, contentDescription = label)  
 Spacer(modifier = Modifier.*width*(16.*dp*))  
 Text(label)  
 **}**}  
  
@Composable  
fun MyScrollContent(padding: PaddingValues) {  
 val scrollState = rememberScrollState()  
 Column(  
 modifier = Modifier  
 .*padding*(padding)  
 .*verticalScroll*(scrollState)  
 .*fillMaxSize*()  
 ) **{** *repeat*(20) **{** index **->** Text(  
 text = "Item $index",  
 modifier = Modifier  
 .*padding*(16.*dp*)  
 .*fillMaxWidth*()  
 )  
 **}  
 }**}  
  
@OptIn(ExperimentalMaterial3Api::class)  
@Composable  
fun ScreenRateUs(navController: NavHostController) {  
 Scaffold(  
 topBar = **{** CenterAlignedTopAppBar(  
 title = **{** Text("Rate Us") **}**,  
 navigationIcon = **{** IconButton(onClick = **{** navController.popBackStack() **}**) **{** Icon(Icons.Filled.*ArrowBack*, contentDescription = "Back")  
 **}  
 }** )  
 **}** ) **{** innerPadding **->** Column(  
 modifier = Modifier  
 .*padding*(innerPadding)  
 .*fillMaxSize*(),  
 horizontalAlignment = Alignment.CenterHorizontally,  
 verticalArrangement = Arrangement.Center  
 ) **{** Text("This is the Rate Us screen")  
 **}  
 }**}  
  
@OptIn(ExperimentalMaterial3Api::class)  
@Composable  
fun ScreenShare(navController: NavHostController) {  
 Scaffold(  
 topBar = **{** CenterAlignedTopAppBar(  
 title = **{** Text("Share") **}**,  
 navigationIcon = **{** IconButton(onClick = **{** navController.popBackStack() **}**) **{** Icon(Icons.Filled.*ArrowBack*, contentDescription = "Back")  
 **}  
 }** )  
 **}** ) **{** innerPadding **->** Column(  
 modifier = Modifier  
 .*padding*(innerPadding)  
 .*fillMaxSize*(),  
 horizontalAlignment = Alignment.CenterHorizontally,  
 verticalArrangement = Arrangement.Center  
 ) **{** Text("This is the Share screen")  
 **}  
 }**}  
  
@OptIn(ExperimentalMaterial3Api::class)  
@Composable  
fun ScreenLogin(navController: NavHostController) {  
 Scaffold(  
 topBar = **{** CenterAlignedTopAppBar(  
 title = **{** Text("Login") **}**,  
 navigationIcon = **{** IconButton(onClick = **{** navController.popBackStack() **}**) **{** Icon(Icons.Filled.*ArrowBack*, contentDescription = "Back")  
 **}  
 }** )  
 **}** ) **{** innerPadding **->** Column(  
 modifier = Modifier  
 .*padding*(innerPadding)  
 .*fillMaxSize*(),  
 horizontalAlignment = Alignment.CenterHorizontally,  
 verticalArrangement = Arrangement.Center  
 ) **{** Text("This is the Login screen")  
 **}  
 }**}  
  
@OptIn(ExperimentalMaterial3Api::class)  
@Composable  
fun ScreenSettings(navController: NavHostController) {  
 Scaffold(  
 topBar = **{** CenterAlignedTopAppBar(  
 title = **{** Text("Settings") **}**,  
 navigationIcon = **{** IconButton(onClick = **{** navController.popBackStack() **}**) **{** Icon(Icons.Filled.*ArrowBack*, contentDescription = "Back")  
 **}  
 }** )  
 **}** ) **{** innerPadding **->** Column(  
 modifier = Modifier  
 .*padding*(innerPadding)  
 .*fillMaxSize*(),  
 horizontalAlignment = Alignment.CenterHorizontally,  
 verticalArrangement = Arrangement.Center  
 ) **{** Text("This is the Settings screen")  
 **}  
 }**}  
  
@OptIn(ExperimentalMaterial3Api::class)  
@Composable  
fun ScreenTrash(navController: NavHostController) {  
 Scaffold(  
 topBar = **{** CenterAlignedTopAppBar(  
 title = **{** Text("Trash") **}**,  
 navigationIcon = **{** IconButton(onClick = **{** navController.popBackStack() **}**) **{** Icon(Icons.Filled.*ArrowBack*, contentDescription = "Back")  
 **}  
 }** )  
 **}** ) **{** innerPadding **->** Column(  
 modifier = Modifier  
 .*padding*(innerPadding)  
 .*fillMaxSize*(),  
 horizontalAlignment = Alignment.CenterHorizontally,  
 verticalArrangement = Arrangement.Center  
 ) **{** Text("This is the Trash screen")  
 **}  
 }**}  
  
@OptIn(ExperimentalMaterial3Api::class)  
@Composable  
fun ScreenSync(navController: NavHostController) {  
 Scaffold(  
 topBar = **{** CenterAlignedTopAppBar(  
 title = **{** Text("Sync") **}**,  
 navigationIcon = **{** IconButton(onClick = **{** navController.popBackStack() **}**) **{** Icon(Icons.Filled.*ArrowBack*, contentDescription = "Back")  
 **}  
 }** )  
 **}** ) **{** innerPadding **->** Column(  
 modifier = Modifier  
 .*padding*(innerPadding)  
 .*fillMaxSize*(),  
 horizontalAlignment = Alignment.CenterHorizontally,  
 verticalArrangement = Arrangement.Center  
 ) **{** Text("This is the Sync screen")  
 **}  
 }**}  
  
@OptIn(ExperimentalMaterial3Api::class)  
@Composable  
fun ScreenMessage(navController: NavHostController) {  
 Scaffold(  
 topBar = **{** CenterAlignedTopAppBar(  
 title = **{** Text("Message") **}**,  
 navigationIcon = **{** IconButton(onClick = **{** navController.popBackStack() **}**) **{** Icon(Icons.Filled.*ArrowBack*, contentDescription = "Back")  
 **}  
 }** )  
 **}** ) **{** innerPadding **->** Column(  
 modifier = Modifier  
 .*padding*(innerPadding)  
 .*fillMaxSize*(),  
 horizontalAlignment = Alignment.CenterHorizontally,  
 verticalArrangement = Arrangement.Center  
 ) **{** Text("This is the Message screen")  
 **}  
 }**}  
  
@OptIn(ExperimentalMaterial3Api::class)  
@Composable  
fun Screen1(navController: NavHostController) {  
 Scaffold(  
 topBar = **{** CenterAlignedTopAppBar(  
 title = **{** Text("Screen 1") **}**,  
 navigationIcon = **{** IconButton(onClick = **{** navController.navigate("main") **}**) **{** Icon(Icons.Filled.*ArrowBack*, contentDescription = "Back")  
 **}  
 }** )  
 **}** ) **{** Column(  
 modifier = Modifier  
 .*fillMaxSize*()  
 .*padding*(**it**),  
 horizontalAlignment = Alignment.CenterHorizontally,  
 verticalArrangement = Arrangement.Center  
 ) **{** Text(text = "This is Screen 1")  
 **}  
 }**}  
  
@OptIn(ExperimentalMaterial3Api::class)  
@Composable  
fun Screen2(navController: NavHostController) {  
 Scaffold(  
 topBar = **{** CenterAlignedTopAppBar(  
 title = **{** Text("Screen 2") **}**,  
 navigationIcon = **{** IconButton(onClick = **{** navController.navigate("main") **}**) **{** Icon(  
 imageVector = Icons.Filled.*ArrowBack*,  
 contentDescription = "Back"  
 )  
 **}  
 }** )  
 **}** ) **{** Column(  
 modifier = Modifier  
 .*fillMaxSize*()  
 .*padding*(**it**),  
 horizontalAlignment = Alignment.CenterHorizontally,  
 verticalArrangement = Arrangement.Center  
 ) **{** Text(text = "This is Screen 2")  
 **}  
 }**}  
  
@Composable  
fun AppBarCenterAlignedTheme(content: @Composable () -> Unit) {  
 MaterialTheme(  
 // Provide custom theme setup if any  
 content = content  
 )  
}

**Sample 8**



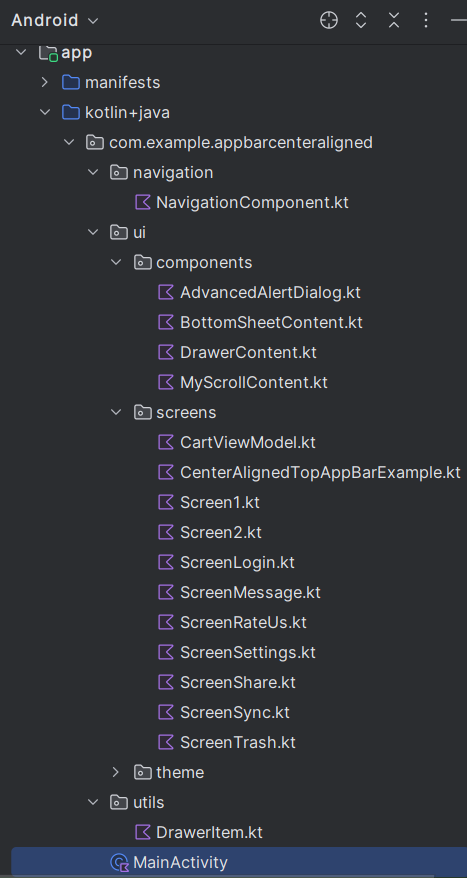


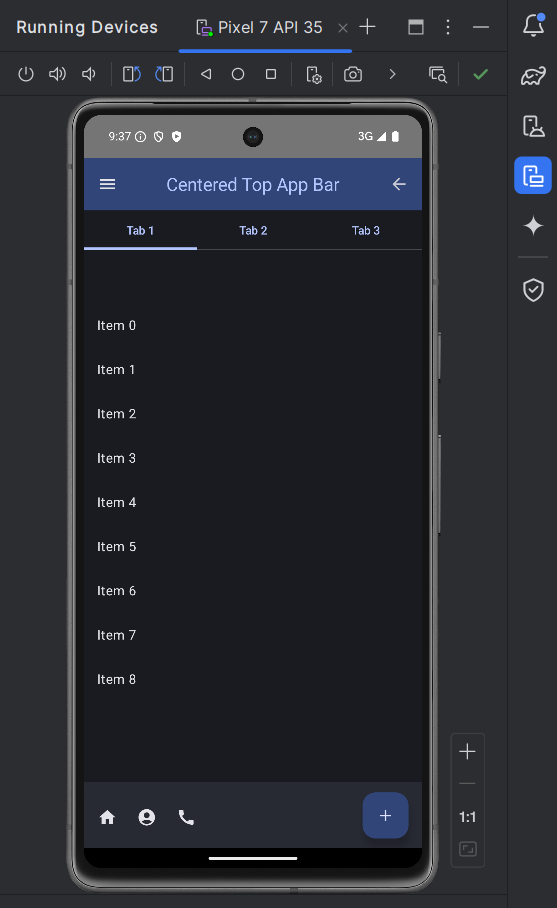


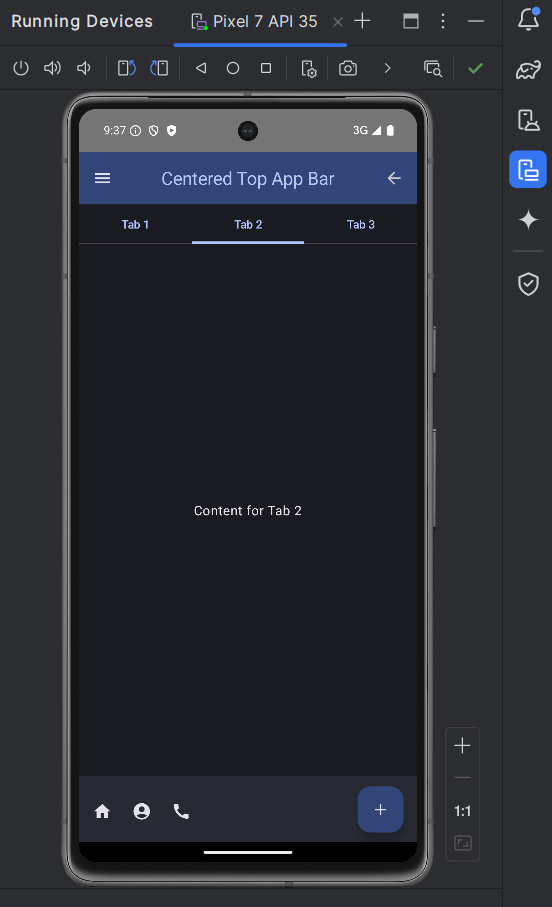


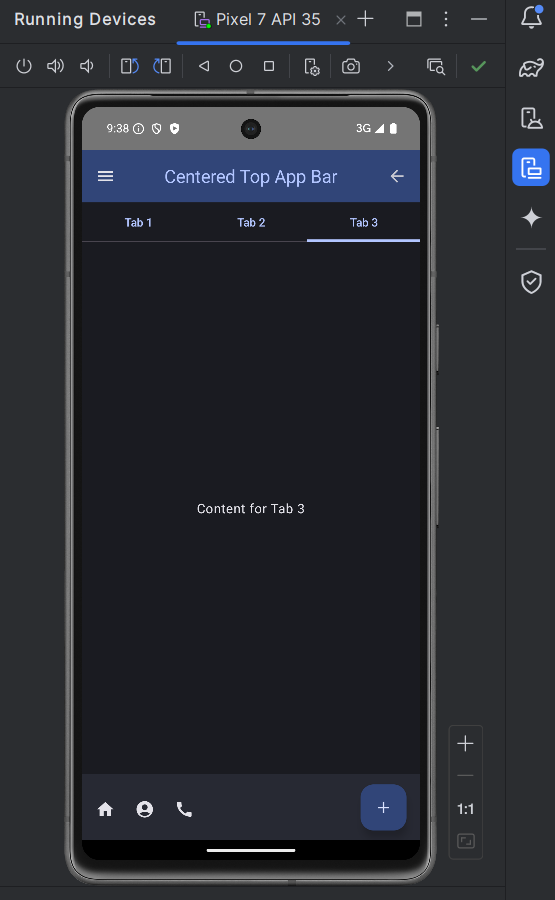
package com.example.simplecard  
  
import android.content.Context  
import android.content.Intent  
import android.net.Uri  
import android.os.Bundle  
import androidx.activity.ComponentActivity  
import androidx.activity.compose.setContent  
import androidx.activity.enableEdgeToEdge  
import androidx.activity.viewModels  
import androidx.compose.foundation.Image  
import androidx.compose.foundation.layout.\*  
import androidx.compose.foundation.lazy.LazyColumn  
import androidx.compose.foundation.lazy.items  
import androidx.compose.foundation.shape.RoundedCornerShape  
import androidx.compose.material.icons.Icons  
import androidx.compose.material.icons.filled.*Delete*import androidx.compose.material.icons.filled.*Favorite*import androidx.compose.material3.\*  
import androidx.compose.runtime.Composable  
import androidx.compose.runtime.collectAsState  
import androidx.compose.ui.Alignment  
import androidx.compose.ui.Modifier  
import androidx.compose.ui.graphics.Color  
import androidx.compose.ui.graphics.painter.Painter  
import androidx.compose.ui.layout.ContentScale  
import androidx.compose.ui.platform.*LocalContext*import androidx.compose.ui.res.painterResource  
import androidx.compose.ui.text.font.FontWeight  
import androidx.compose.ui.unit.dp  
import androidx.compose.ui.unit.sp  
import androidx.lifecycle.ViewModel  
import androidx.lifecycle.*viewModelScope*import androidx.navigation.NavHostController  
import androidx.navigation.compose.NavHost  
import androidx.navigation.compose.composable  
import androidx.navigation.compose.rememberNavController  
import com.example.simplecard.ui.theme.SimpleCardTheme  
import kotlinx.coroutines.flow.MutableStateFlow  
import kotlinx.coroutines.flow.StateFlow  
import kotlinx.coroutines.launch  
  
data class CartItem(val title: String, val subtitle: String, val imageRes: Int)  
  
class CartViewModel : ViewModel() {  
 private val \_cartItems = *MutableStateFlow*<List<CartItem>>(*emptyList*())  
 val cartItems: StateFlow<List<CartItem>> = \_cartItems  
  
 fun addItem(item: CartItem) {  
 *viewModelScope*.*launch* **{** \_cartItems.value = \_cartItems.value + item  
 **}** }  
  
 fun removeItem(item: CartItem) {  
 *viewModelScope*.*launch* **{** \_cartItems.value = \_cartItems.value - item  
 **}** }  
}  
  
class MainActivity : ComponentActivity() {  
 private val cartViewModel: CartViewModel by *viewModels*()  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 *enableEdgeToEdge*()  
 *setContent* **{** SimpleCardTheme **{** val navController = rememberNavController()  
 Scaffold(modifier = Modifier.*fillMaxSize*()) **{** innerPadding **->** NavHost(  
 navController = navController,  
 startDestination = "cardList",  
 modifier = Modifier.*padding*(innerPadding)  
 ) **{** *composable*("cardList") **{** CardListScreen(navController, cartViewModel)  
 **}** *composable*("cartScreen") **{** val cartItems = cartViewModel.cartItems.collectAsState().value  
 CartScreen(navController, cartItems, cartViewModel)  
 **}  
 }  
 }  
 }  
 }** }  
}  
  
@Composable  
fun CardListScreen(navController: NavHostController, cartViewModel: CartViewModel) {  
 LazyColumn(  
 modifier = Modifier  
 .*fillMaxSize*()  
 ) **{** item **{** ComplexCard1(onAddToCart = **{** cartViewModel.addItem(CartItem("Hamburger1 Title1", "Ingredient1 Subtitle1", R.drawable.*\_d\_burger*))  
 navController.navigate("cartScreen")  
 **}**)  
 **}** item **{** ComplexCard2(onAddToCart = **{** cartViewModel.addItem(CartItem("Hamburger2 Title2", "Ingredient2 Subtitle2", R.drawable.*burger1*))  
 navController.navigate("cartScreen")  
 **}**)  
 **}** item **{** ComplexCard3(onAddToCart = **{** cartViewModel.addItem(CartItem("Hamburger3 Title3", "Ingredient3 Subtitle3", R.drawable.*\_d\_burger*))  
 navController.navigate("cartScreen")  
 **}**)  
 **}** item **{** ComplexCard4(onAddToCart = **{** cartViewModel.addItem(CartItem("Hamburger4 Title4", "Ingredient4 Subtitle4", R.drawable.*\_d\_burger*))  
 navController.navigate("cartScreen")  
 **}**)  
 **}** item **{** ComplexCard5(onAddToCart = **{** cartViewModel.addItem(CartItem("Hamburger5 Title5", "Ingredient5 Subtitle5", R.drawable.*\_d\_burger*))  
 navController.navigate("cartScreen")  
 **}**)  
 **}  
 }**}  
  
@Composable  
fun ComplexCard1(onAddToCart: () -> Unit, modifier: Modifier = Modifier) {  
 Card(  
 modifier = modifier  
 .*padding*(16.*dp*)  
 .*fillMaxWidth*(),  
 shape = *RoundedCornerShape*(16.*dp*),  
 elevation = CardDefaults.cardElevation(8.*dp*)  
 ) **{** Column **{** val image: Painter = painterResource(id = R.drawable.*\_d\_burger*)  
 Image(  
 painter = image,  
 contentDescription = null,  
 contentScale = ContentScale.Crop,  
 modifier = Modifier  
 .*height*(300.*dp*)  
 .*fillMaxWidth*()  
 )  
 Column(modifier = Modifier.*padding*(16.*dp*)) **{** Text(text = "Card1 Title1", fontSize = 20.*sp*, fontWeight = FontWeight.Bold)  
 Spacer(modifier = Modifier.*height*(8.*dp*))  
 Text(text = "Card1 Subtitle1", fontSize = 16.*sp*, color = Color.Gray)  
 Spacer(modifier = Modifier.*height*(8.*dp*))  
 Text(  
 text = "This is a description for the card1. It can be multiple lines long and should give a brief overview of the content.",  
 fontSize = 14.*sp* )  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
 Row(  
 verticalAlignment = Alignment.CenterVertically  
 ) **{** Button(onClick = onAddToCart) **{** Text("Add to cart")  
 **}** Spacer(modifier = Modifier.*width*(8.*dp*))  
 IconButton(onClick = **{** /\* *TODO* \*/ **}**) **{** Icon(Icons.Filled.*Favorite*, contentDescription = "Favorite")  
 **}  
 }  
 }  
 }  
 }**}  
  
@Composable  
fun ComplexCard2(onAddToCart: () -> Unit, modifier: Modifier = Modifier) {  
 Card(  
 modifier = modifier  
 .*padding*(16.*dp*)  
 .*fillMaxWidth*(),  
 shape = *RoundedCornerShape*(16.*dp*),  
 elevation = CardDefaults.cardElevation(8.*dp*)  
 ) **{** Column **{** val image: Painter = painterResource(id = R.drawable.*burger1*)  
 Image(  
 painter = image,  
 contentDescription = null,  
 contentScale = ContentScale.Crop,  
 modifier = Modifier  
 .*height*(350.*dp*)  
 .*fillMaxWidth*()  
 )  
 Column(modifier = Modifier.*padding*(16.*dp*)) **{** Text(text = "Card Title", fontSize = 20.*sp*, fontWeight = FontWeight.Bold)  
 Spacer(modifier = Modifier.*height*(8.*dp*))  
 Text(text = "Card Subtitle", fontSize = 16.*sp*, color = Color.Gray)  
 Spacer(modifier = Modifier.*height*(8.*dp*))  
 Text(  
 text = "This is a description for the card. It can be multiple lines long and should give a brief overview of the content.",  
 fontSize = 14.*sp* )  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
 Row(  
 verticalAlignment = Alignment.CenterVertically  
 ) **{** Button(onClick = onAddToCart) **{** Text("Add to cart")  
 **}** Spacer(modifier = Modifier.*width*(8.*dp*))  
 IconButton(onClick = **{** /\* *TODO* \*/ **}**) **{** Icon(Icons.Filled.*Favorite*, contentDescription = "Favorite")  
 **}  
 }  
 }  
 }  
 }**}  
  
@Composable  
fun ComplexCard3(onAddToCart: () -> Unit, modifier: Modifier = Modifier) {  
 Card(  
 modifier = modifier  
 .*padding*(16.*dp*)  
 .*fillMaxWidth*(),  
 shape = *RoundedCornerShape*(16.*dp*),  
 elevation = CardDefaults.cardElevation(8.*dp*)  
 ) **{** Column **{** val image: Painter = painterResource(id = R.drawable.*\_d\_burger*)  
 Image(  
 painter = image,  
 contentDescription = null,  
 contentScale = ContentScale.Crop,  
 modifier = Modifier  
 .*height*(300.*dp*)  
 .*fillMaxWidth*()  
 )  
 Column(modifier = Modifier.*padding*(16.*dp*)) **{** Text(text = "Card Title", fontSize = 20.*sp*, fontWeight = FontWeight.Bold)  
 Spacer(modifier = Modifier.*height*(8.*dp*))  
 Text(text = "Card Subtitle", fontSize = 16.*sp*, color = Color.Gray)  
 Spacer(modifier = Modifier.*height*(8.*dp*))  
 Text(  
 text = "This is a description for the card. It can be multiple lines long and should give a brief overview of the content.",  
 fontSize = 14.*sp* )  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
 Row(  
 verticalAlignment = Alignment.CenterVertically  
 ) **{** Button(onClick = onAddToCart) **{** Text("Add to cart")  
 **}** Spacer(modifier = Modifier.*width*(8.*dp*))  
 IconButton(onClick = **{** /\* *TODO* \*/ **}**) **{** Icon(Icons.Filled.*Favorite*, contentDescription = "Favorite")  
 **}  
 }  
 }  
 }  
 }**}  
  
@Composable  
fun ComplexCard4(onAddToCart: () -> Unit, modifier: Modifier = Modifier) {  
 Card(  
 modifier = modifier  
 .*padding*(16.*dp*)  
 .*fillMaxWidth*(),  
 shape = *RoundedCornerShape*(16.*dp*),  
 elevation = CardDefaults.cardElevation(8.*dp*)  
 ) **{** Column **{** val image: Painter = painterResource(id = R.drawable.*\_d\_burger*)  
 Image(  
 painter = image,  
 contentDescription = null,  
 contentScale = ContentScale.Crop,  
 modifier = Modifier  
 .*height*(300.*dp*)  
 .*fillMaxWidth*()  
 )  
 Column(modifier = Modifier.*padding*(16.*dp*)) **{** Text(text = "Card Title", fontSize = 20.*sp*, fontWeight = FontWeight.Bold)  
 Spacer(modifier = Modifier.*height*(8.*dp*))  
 Text(text = "Card Subtitle", fontSize = 16.*sp*, color = Color.Gray)  
 Spacer(modifier = Modifier.*height*(8.*dp*))  
 Text(  
 text = "This is a description for the card. It can be multiple lines long and should give a brief overview of the content.",  
 fontSize = 14.*sp* )  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
 Row(  
 verticalAlignment = Alignment.CenterVertically  
 ) **{** Button(onClick = onAddToCart) **{** Text("Add to cart")  
 **}** Spacer(modifier = Modifier.*width*(8.*dp*))  
 IconButton(onClick = **{** /\* *TODO* \*/ **}**) **{** Icon(Icons.Filled.*Favorite*, contentDescription = "Favorite")  
 **}  
 }  
 }  
 }  
 }**}  
  
@Composable  
fun ComplexCard5(onAddToCart: () -> Unit, modifier: Modifier = Modifier) {  
 Card(  
 modifier = modifier  
 .*padding*(16.*dp*)  
 .*fillMaxWidth*(),  
 shape = *RoundedCornerShape*(16.*dp*),  
 elevation = CardDefaults.cardElevation(8.*dp*)  
 ) **{** Column **{** val image: Painter = painterResource(id = R.drawable.*\_d\_burger*)  
 Image(  
 painter = image,  
 contentDescription = null,  
 contentScale = ContentScale.Crop,  
 modifier = Modifier  
 .*height*(300.*dp*)  
 .*fillMaxWidth*()  
 )  
 Column(modifier = Modifier.*padding*(16.*dp*)) **{** Text(text = "Card Title", fontSize = 20.*sp*, fontWeight = FontWeight.Bold)  
 Spacer(modifier = Modifier.*height*(8.*dp*))  
 Text(text = "Card Subtitle", fontSize = 16.*sp*, color = Color.Gray)  
 Spacer(modifier = Modifier.*height*(8.*dp*))  
 Text(  
 text = "This is a description for the card. It can be multiple lines long and should give a brief overview of the content.",  
 fontSize = 14.*sp* )  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
 Row(  
 verticalAlignment = Alignment.CenterVertically  
 ) **{** Button(onClick = onAddToCart) **{** Text("Add to cart")  
 **}** Spacer(modifier = Modifier.*width*(8.*dp*))  
 IconButton(onClick = **{** /\* *TODO* \*/ **}**) **{** Icon(Icons.Filled.*Favorite*, contentDescription = "Favorite")  
 **}  
 }  
 }  
 }  
 }**}  
  
@Composable  
fun CartScreen(navController: NavHostController, cartItems: List<CartItem>, cartViewModel: CartViewModel, modifier: Modifier = Modifier) {  
 val context = *LocalContext*.current  
  
 Column(modifier = modifier.*padding*(16.*dp*)) **{** Button(onClick = **{** navController.navigate("cardList") **}**) **{** Text("Back to main screen")  
 **}** Spacer(modifier = Modifier.*height*(16.*dp*))  
 LazyColumn **{** *items*(cartItems) **{** item **->** Card(  
 modifier = Modifier  
 .*padding*(vertical = 8.*dp*)  
 .*fillMaxWidth*(),  
 shape = *RoundedCornerShape*(16.*dp*),  
 elevation = CardDefaults.cardElevation(8.*dp*)  
 ) **{** Row(modifier = Modifier.*padding*(16.*dp*), verticalAlignment = Alignment.CenterVertically) **{** Image(  
 painter = painterResource(id = item.imageRes),  
 contentDescription = null,  
 modifier = Modifier.*size*(64.*dp*),  
 contentScale = ContentScale.Crop  
 )  
 Spacer(modifier = Modifier.*width*(16.*dp*))  
 Column(  
 modifier = Modifier.*weight*(1f)  
 ) **{** Text(text = item.title, fontSize = 20.*sp*, fontWeight = FontWeight.Bold)  
 Spacer(modifier = Modifier.*height*(8.*dp*))  
 Text(text = item.subtitle, fontSize = 16.*sp*, color = Color.Gray)  
 **}** IconButton(onClick = **{** cartViewModel.removeItem(item) **}**) **{** Icon(Icons.Default.*Delete*, contentDescription = "Remove item")  
 **}  
 }  
 }  
 }  
 }** Spacer(modifier = Modifier.*height*(16.*dp*))  
 Button(onClick = **{** *submitOrder*(cartItems, context) **}**) **{** Text("Submit Order")  
 **}  
 }**}  
  
fun submitOrder(cartItems: List<CartItem>, context: Context) {  
 val message = *buildString* **{** append("Order details:\n")  
 cartItems.*forEach* **{** item **->** append("Title: ${item.title}, Subtitle: ${item.subtitle}\n")  
 **}  
 }** *sendChatMessageToCooker*(message, context)  
}  
  
fun sendChatMessageToCooker(message: String, context: Context) {  
 *sendWhatsAppMessage*(context, "+34607770997", message)  
}  
  
fun sendWhatsAppMessage(context: Context, phoneNumber: String, message: String) {  
 val intent = Intent(Intent.*ACTION\_VIEW*).*apply* **{** *data* = Uri.parse("https://wa.me/$phoneNumber?text=${Uri.encode(message)}")  
 **}** context.startActivity(intent)  
}

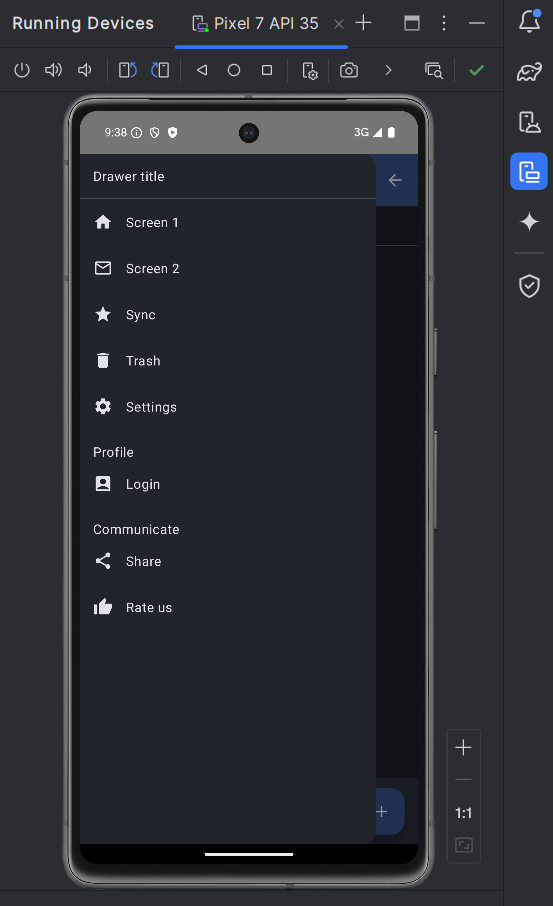
**Sample 9**

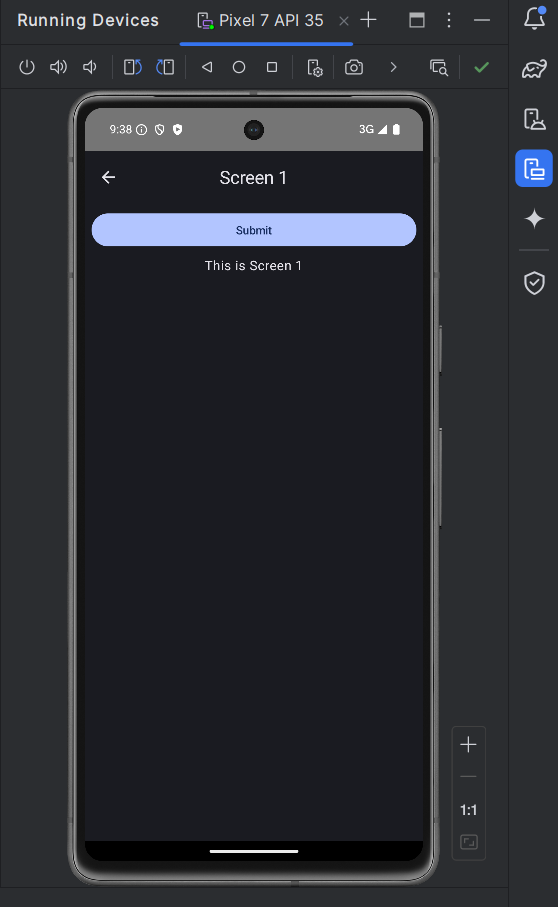


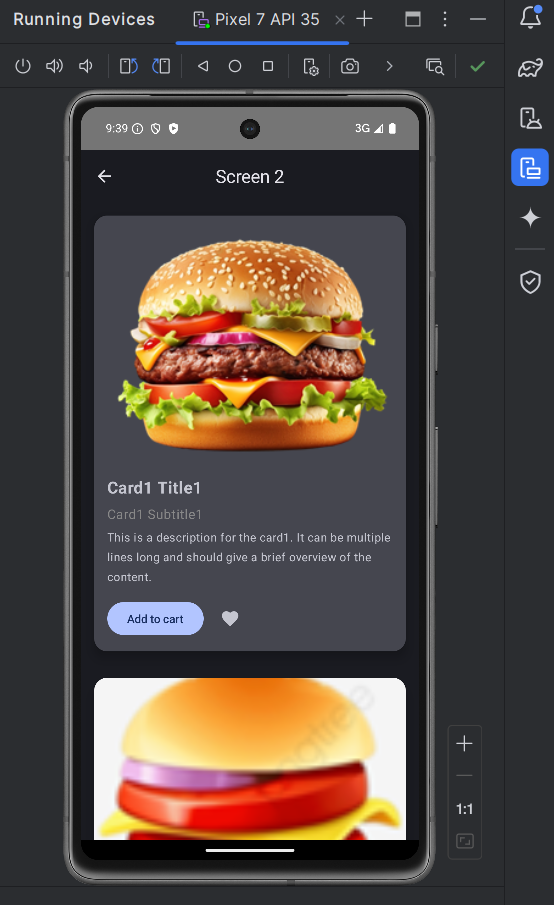


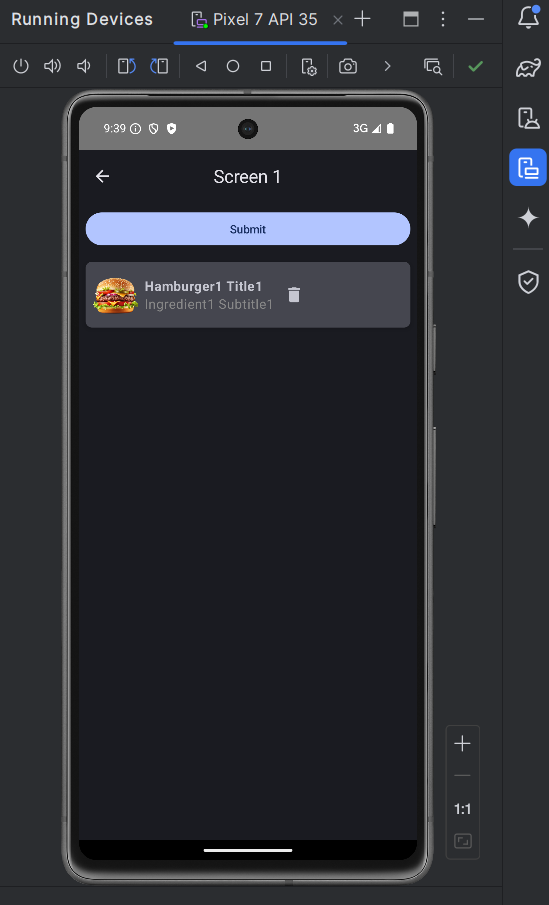


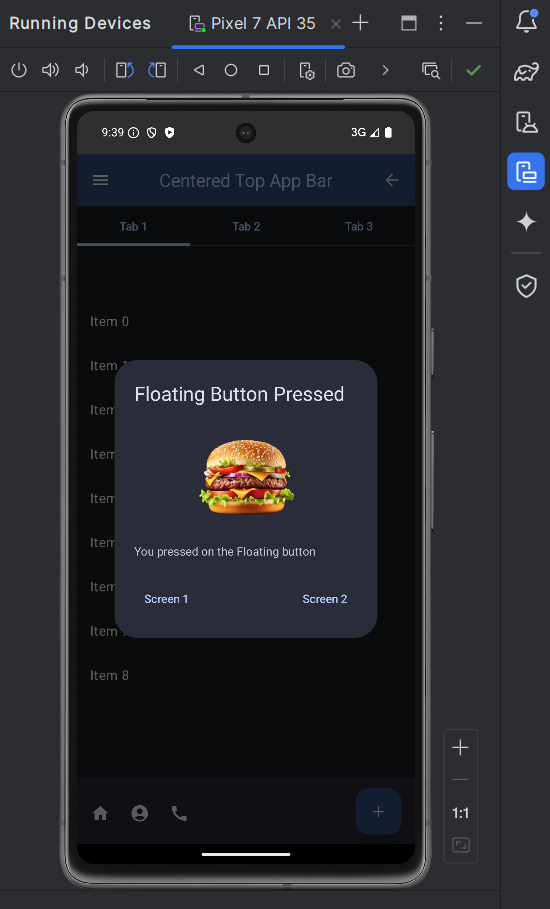


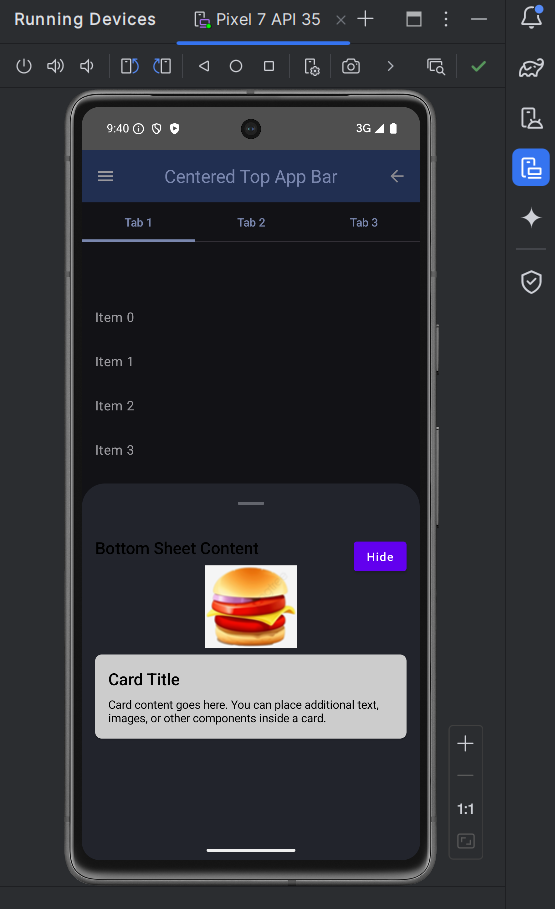


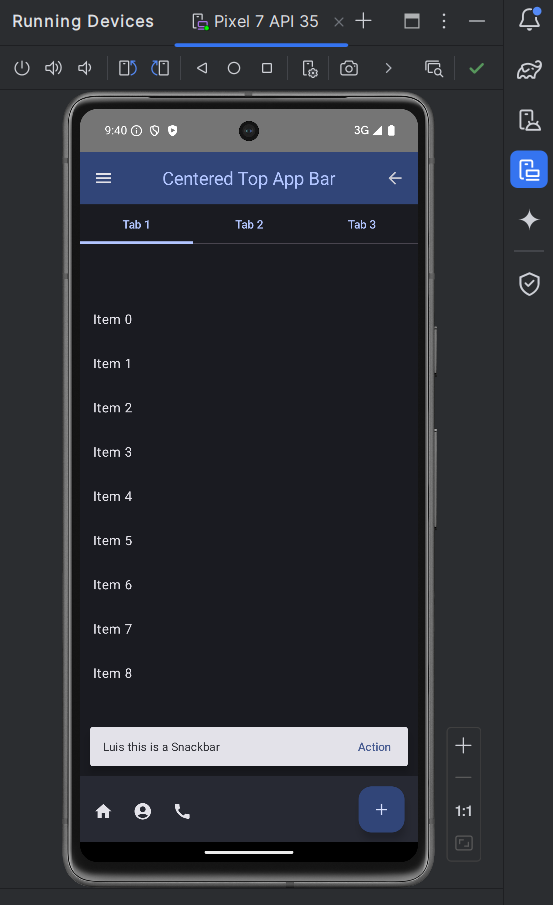












**MainActivity.kt**

package com.example.appbarcenteraligned  
  
import android.os.Bundle  
import androidx.activity.ComponentActivity  
import androidx.activity.compose.setContent  
import androidx.navigation.compose.rememberNavController  
import com.example.appbarcenteraligned.navigation.NavigationComponent  
import com.example.appbarcenteraligned.ui.theme.AppBarCenterAlignedTheme  
  
class MainActivity : ComponentActivity() {  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 *setContent* **{** AppBarCenterAlignedTheme **{** val navController = rememberNavController()  
 NavigationComponent(navController)  
 **}  
 }** }  
}

**navigation/NavigationComponent.kt**

package com.example.appbarcenteraligned.navigation  
  
import androidx.compose.runtime.Composable  
import androidx.lifecycle.viewmodel.compose.viewModel  
import androidx.navigation.NavHostController  
import androidx.navigation.compose.NavHost  
import androidx.navigation.compose.composable  
import com.example.appbarcenteraligned.ui.screens.\*  
  
@Composable  
fun NavigationComponent(navController: NavHostController) {  
 val cartViewModel: CartViewModel = viewModel()  
  
 NavHost(navController, startDestination = "main") **{** *composable*("main") **{** CenterAlignedTopAppBarExample(navController) **}** *composable*("screen1") **{** Screen1(navController, cartViewModel) **}** *composable*("screen2") **{** Screen2(navController, cartViewModel) **}** *composable*("message") **{** ScreenMessage(navController) **}** *composable*("sync") **{** ScreenSync(navController) **}** *composable*("trash") **{** ScreenTrash(navController) **}** *composable*("settings") **{** ScreenSettings(navController) **}** *composable*("login") **{** ScreenLogin(navController) **}** *composable*("share") **{** ScreenShare(navController) **}** *composable*("rate\_us") **{** ScreenRateUs(navController) **}  
 }**}

**screens/CenterAlignedTopAppBarExample.kt**

package com.example.appbarcenteraligned.ui.screens  
  
import androidx.compose.foundation.ExperimentalFoundationApi  
import androidx.compose.foundation.layout.\*  
import androidx.compose.foundation.pager.HorizontalPager  
import androidx.compose.foundation.pager.rememberPagerState  
import androidx.compose.material.ExperimentalMaterialApi  
import androidx.compose.material.ModalBottomSheetValue  
import androidx.compose.material.icons.Icons  
import androidx.compose.material.icons.filled.\*  
import androidx.compose.material3.\*  
import androidx.compose.runtime.\*  
import androidx.compose.ui.Alignment  
import androidx.compose.ui.Modifier  
import androidx.compose.ui.input.nestedscroll.nestedScroll  
import androidx.compose.ui.res.painterResource  
import androidx.compose.ui.text.style.TextOverflow  
import androidx.navigation.NavHostController  
import com.example.appbarcenteraligned.R  
import com.example.appbarcenteraligned.ui.components.\*  
import com.google.accompanist.pager.\*  
import kotlinx.coroutines.launch  
  
@OptIn(ExperimentalMaterial3Api::class, ExperimentalMaterialApi::class, ExperimentalPagerApi::class,  
 ExperimentalFoundationApi::class  
)  
@Composable  
fun CenterAlignedTopAppBarExample(navController: NavHostController) {  
 val drawerState = rememberDrawerState(DrawerValue.*Closed*)  
 val coroutineScope = rememberCoroutineScope()  
 val scrollBehavior = TopAppBarDefaults.pinnedScrollBehavior(rememberTopAppBarState())  
 val showDialog = remember **{** *mutableStateOf*(false) **}** val snackbarHostState = remember **{** SnackbarHostState() **}** val scope = rememberCoroutineScope()  
 val imagePainter = painterResource(id = R.drawable.*\_d\_burger*)  
  
 if (showDialog.value) {  
 AdvancedAlertDialog(  
 showDialog = showDialog,  
 onNavigateToScreen1 = **{** navController.navigate("screen1") **}**,  
 onNavigateToScreen2 = **{** navController.navigate("screen2") **}**,  
 imagePainter = imagePainter  
 )  
 }  
  
 var showBottomSheet by remember **{** *mutableStateOf*(false) **}** val sheetState = rememberModalBottomSheetState(false)  
 val pagerState = rememberPagerState()  
  
 ModalNavigationDrawer(  
 drawerState = drawerState,  
 drawerContent = **{** DrawerContent(navController, drawerState, coroutineScope)  
 **}** ) **{** Scaffold(  
 modifier = Modifier.*nestedScroll*(scrollBehavior.nestedScrollConnection),  
 topBar = **{** CenterAlignedTopAppBar(  
 colors = TopAppBarDefaults.topAppBarColors(  
 containerColor = MaterialTheme.colorScheme.primaryContainer,  
 titleContentColor = MaterialTheme.colorScheme.primary,  
 ),  
 title = **{** Text(  
 "Centered Top App Bar",  
 maxLines = 1,  
 overflow = TextOverflow.Ellipsis  
 )  
 **}**,  
 navigationIcon = **{** IconButton(onClick = **{** coroutineScope.*launch* **{** drawerState.open() **} }**) **{** Icon(  
 imageVector = Icons.Filled.*Menu*,  
 contentDescription = "Menu"  
 )  
 **}  
 }**,  
 actions = **{** IconButton(onClick = **{** navController.popBackStack() **}**) **{** Icon(  
 imageVector = Icons.Filled.*ArrowBack*,  
 contentDescription = "Back"  
 )  
 **}  
 }**,  
 scrollBehavior = scrollBehavior,  
 )  
 **}**,  
 bottomBar = **{** BottomAppBar(  
 actions = **{** IconButton(onClick = **{** scope.*launch* **{** val result = snackbarHostState.showSnackbar(  
 message = "Luis this is a Snackbar",  
 actionLabel = "Action",  
 duration = SnackbarDuration.*Indefinite* )  
 when (result) {  
 SnackbarResult.*ActionPerformed* -> {  
 navController.navigate("screen1")  
 }  
 SnackbarResult.*Dismissed* -> {  
 }  
 }  
 **}  
 }**) **{** Icon(imageVector = Icons.Filled.*Home*, contentDescription = "Button 1")  
 **}** IconButton(onClick = **{** showBottomSheet = true  
 **}**) **{** Icon(imageVector = Icons.Filled.*AccountCircle*, contentDescription = "Button 2")  
 **}** IconButton(onClick = **{ }**) **{** Icon(imageVector = Icons.Filled.*Call*, contentDescription = "Button 3")  
 **}  
 }**,  
 floatingActionButton = **{** FloatingActionButton(onClick = **{** showDialog.value = true **}**) **{** Icon(imageVector = Icons.Filled.*Add*, contentDescription = "Floating Button")  
 **}  
 }** )  
 **}**,  
 floatingActionButtonPosition = FabPosition.End,  
 snackbarHost = **{** SnackbarHost(hostState = snackbarHostState)  
 **}** ) **{** innerPadding **->** Column(modifier = Modifier.*padding*(innerPadding).*fillMaxSize*()) **{** val tabTitles = *listOf*("Tab 1", "Tab 2", "Tab 3")  
  
 TabRow(  
 selectedTabIndex = pagerState.currentPage,  
 modifier = Modifier.*fillMaxWidth*()  
 ) **{** tabTitles.*forEachIndexed* **{** index, title **->** Tab(  
 selected = pagerState.currentPage == index,  
 onClick = **{** coroutineScope.*launch* **{** pagerState.scrollToPage(index) **} }**,  
 text = **{** Text(title) **}** )  
 **}  
 }** HorizontalPager(  
 count = tabTitles.size,  
 state = pagerState,  
 modifier = Modifier.*weight*(1f)  
 ) **{** page **->** when (page) {  
 0 -> MyScrollContent(innerPadding)  
 1 -> TabContent2()  
 2 -> TabContent3()  
 }  
 **}** if (showBottomSheet) {  
 ModalBottomSheet(  
 onDismissRequest = **{** showBottomSheet = false  
 **}**,  
 sheetState = sheetState  
 ) **{** BottomSheetContent(onHide = **{** scope.*launch* **{** sheetState.hide() **}**.invokeOnCompletion **{** if (!sheetState.isVisible) {  
 showBottomSheet = false  
 }  
 **}  
 }**)  
 **}** }  
 **}  
 }  
 }**}  
  
@Composable  
fun TabContent2() {  
 Box(modifier = Modifier.*fillMaxSize*(), contentAlignment = Alignment.Center) **{** Text(text = "Content for Tab 2")  
 **}**}  
  
@Composable  
fun TabContent3() {  
 Box(modifier = Modifier.*fillMaxSize*(), contentAlignment = Alignment.Center) **{** Text(text = "Content for Tab 3")  
 **}**}

**screens/Screen1.kt**

package com.example.appbarcenteraligned.ui.screens  
  
import android.content.Context  
import android.content.Intent  
import android.net.Uri  
import androidx.compose.foundation.Image  
import androidx.compose.foundation.layout.\*  
import androidx.compose.foundation.lazy.LazyColumn  
import androidx.compose.foundation.lazy.items  
import androidx.compose.foundation.shape.RoundedCornerShape  
import androidx.compose.material.icons.Icons  
import androidx.compose.material.icons.filled.*ArrowBack*import androidx.compose.material.icons.filled.*Delete*import androidx.compose.material3.\*  
import androidx.compose.runtime.Composable  
import androidx.compose.runtime.collectAsState  
import androidx.compose.runtime.getValue  
import androidx.compose.ui.Alignment  
import androidx.compose.ui.Modifier  
import androidx.compose.ui.graphics.Color  
import androidx.compose.ui.layout.ContentScale  
import androidx.compose.ui.platform.*LocalContext*import androidx.compose.ui.res.painterResource  
import androidx.compose.ui.text.font.FontWeight  
import androidx.compose.ui.unit.dp  
import androidx.lifecycle.viewmodel.compose.viewModel  
import androidx.navigation.NavHostController  
  
@OptIn(ExperimentalMaterial3Api::class)  
@Composable  
fun Screen1(navController: NavHostController, cartViewModel: CartViewModel = viewModel()) {  
 val cartItems by cartViewModel.cartItems.collectAsState()  
  
 val context = *LocalContext*.current  
  
 Scaffold(  
 topBar = **{** CenterAlignedTopAppBar(  
 title = **{** Text("Screen 1") **}**,  
 navigationIcon = **{** IconButton(onClick = **{** navController.navigate("main") **}**) **{** Icon(Icons.Filled.*ArrowBack*, contentDescription = "Back")  
 **}  
 }** )  
 **}** ) **{** Column(  
 modifier = Modifier  
 .*fillMaxSize*()  
 .*padding*(**it**),  
 horizontalAlignment = Alignment.CenterHorizontally,  
 verticalArrangement = Arrangement.Top  
 ) **{** Button(  
 onClick = **{** *submitOrder*(cartItems, context)  
 **}**,  
 modifier = Modifier  
 .*padding*(8.*dp*)  
 .*fillMaxWidth*()  
 ) **{** Text(text = "Submit")  
 **}** if (cartItems.isEmpty()) {  
 Text(text = "This is Screen 1")  
 } else {  
 LazyColumn **{** *items*(cartItems) **{** cartItem **->** CartItemRow(  
 cartItem = cartItem,  
 onDelete = **{** cartViewModel.removeItem(cartItem) **}** )  
 **}  
 }** }  
 **}  
 }**}  
  
fun submitOrder(cartItems: List<CartItem>, context: Context) {  
 val message = *buildString* **{** append("Order details:\n")  
 cartItems.*forEach* **{** item **->** append("Title: ${item.title}, Subtitle: ${item.subtitle}\n")  
 **}  
 }** *sendChatMessageToCooker*(message, context)  
}  
  
fun sendChatMessageToCooker(message: String, context: Context) {  
 *sendWhatsAppMessage*(context, "+34607770997", message)  
}  
  
fun sendWhatsAppMessage(context: Context, phoneNumber: String, message: String) {  
 val intent = Intent(Intent.*ACTION\_VIEW*).*apply* **{** *data* = Uri.parse("https://wa.me/$phoneNumber?text=${Uri.encode(message)}")  
 **}** context.startActivity(intent)  
}  
  
@Composable  
fun CartItemRow(cartItem: CartItem, onDelete: () -> Unit) {  
 Card(  
 modifier = Modifier  
 .*padding*(8.*dp*)  
 .*fillMaxWidth*(),  
 shape = *RoundedCornerShape*(8.*dp*),  
 elevation = CardDefaults.cardElevation(4.*dp*)  
 ) **{** Row(  
 modifier = Modifier.*padding*(8.*dp*),  
 verticalAlignment = Alignment.CenterVertically,  
 horizontalArrangement = Arrangement.SpaceBetween // Ensures elements are spaced out  
 ) **{** Row(  
 verticalAlignment = Alignment.CenterVertically  
 ) **{** Image(  
 painter = painterResource(id = cartItem.imageRes),  
 contentDescription = null,  
 contentScale = ContentScale.Crop,  
 modifier = Modifier  
 .*size*(64.*dp*)  
 .*padding*(end = 8.*dp*)  
 )  
 Column **{** Text(cartItem.title, fontWeight = FontWeight.Bold)  
 Text(cartItem.subtitle, color = Color.Gray)  
 **}  
 }** IconButton(onClick = onDelete) **{** Icon(Icons.Default.*Delete*, contentDescription = "Delete")  
 **}  
 }  
 }**}

**screens/Screen2.kt**

package com.example.appbarcenteraligned.ui.screens  
  
import androidx.compose.foundation.Image  
import androidx.compose.foundation.layout.\*  
import androidx.compose.foundation.lazy.LazyColumn  
import androidx.compose.foundation.lazy.items  
import androidx.compose.foundation.shape.RoundedCornerShape  
import androidx.compose.material.icons.Icons  
import androidx.compose.material.icons.filled.*ArrowBack*import androidx.compose.material.icons.filled.*Favorite*import androidx.compose.material3.\*  
import androidx.compose.runtime.\*  
import androidx.compose.ui.Modifier  
import androidx.compose.ui.Alignment  
import androidx.compose.ui.graphics.Color  
import androidx.compose.ui.graphics.painter.Painter  
import androidx.compose.ui.layout.ContentScale  
import androidx.compose.ui.res.painterResource  
import androidx.compose.ui.text.font.FontWeight  
import androidx.compose.ui.unit.dp  
import androidx.compose.ui.unit.sp  
import androidx.lifecycle.viewmodel.compose.viewModel  
import androidx.navigation.NavHostController  
import com.example.appbarcenteraligned.R  
import kotlinx.coroutines.flow.MutableStateFlow  
import kotlinx.coroutines.flow.StateFlow  
import kotlinx.coroutines.launch  
  
@OptIn(ExperimentalMaterial3Api::class)  
@Composable  
fun Screen2(navController: NavHostController, cartViewModel: CartViewModel) {  
 val cartItems by cartViewModel.cartItems.collectAsState()  
  
 Scaffold(  
 topBar = **{** CenterAlignedTopAppBar(  
 title = **{** Text("Screen 2") **}**,  
 navigationIcon = **{** IconButton(onClick = **{** navController.navigate("main") **}**) **{** Icon(  
 imageVector = Icons.Filled.*ArrowBack*,  
 contentDescription = "Back"  
 )  
 **}  
 }** )  
 **}** ) **{** paddingValues **->** LazyColumn(  
 modifier = Modifier  
 .*fillMaxSize*()  
 .*padding*(paddingValues)  
 ) **{** item **{** ComplexCard1(onAddToCart = **{** cartViewModel.addItem(CartItem("Hamburger1 Title1", "Ingredient1 Subtitle1", R.drawable.*\_d\_burger*))  
 navController.navigate("screen1")  
 **}**)  
 **}** item **{** ComplexCard2(onAddToCart = **{** cartViewModel.addItem(CartItem("Hamburger1 Title1", "Ingredient1 Subtitle1", R.drawable.*burger2*))  
 navController.navigate("screen1")  
 **}**)  
 **}** item **{** ComplexCard3(onAddToCart = **{** cartViewModel.addItem(CartItem("Hamburger1 Title1", "Ingredient1 Subtitle1", R.drawable.*\_d\_burger*))  
 navController.navigate("screen1")  
 **}**)  
 **}  
 }  
 }**}  
  
@Composable  
fun ComplexCard1(onAddToCart: () -> Unit, modifier: Modifier = Modifier) {  
 Card(  
 modifier = modifier  
 .*padding*(16.*dp*)  
 .*fillMaxWidth*(),  
 shape = *RoundedCornerShape*(16.*dp*),  
 elevation = CardDefaults.cardElevation(8.*dp*)  
 ) **{** Column **{** val image: Painter = painterResource(id = R.drawable.*\_d\_burger*)  
 Image(  
 painter = image,  
 contentDescription = null,  
 contentScale = ContentScale.Crop,  
 modifier = Modifier  
 .*height*(300.*dp*)  
 .*fillMaxWidth*()  
 )  
 Column(modifier = Modifier.*padding*(16.*dp*)) **{** Text(text = "Card1 Title1", fontSize = 20.*sp*, fontWeight = FontWeight.Bold)  
 Spacer(modifier = Modifier.*height*(8.*dp*))  
 Text(text = "Card1 Subtitle1", fontSize = 16.*sp*, color = Color.Gray)  
 Spacer(modifier = Modifier.*height*(8.*dp*))  
 Text(  
 text = "This is a description for the card1. It can be multiple lines long and should give a brief overview of the content.",  
 fontSize = 14.*sp* )  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
 Row(  
 verticalAlignment = Alignment.CenterVertically  
 ) **{** Button(onClick = onAddToCart) **{** Text("Add to cart")  
 **}** Spacer(modifier = Modifier.*width*(8.*dp*))  
 IconButton(onClick = **{** /\* *TODO* \*/ **}**) **{** Icon(Icons.Filled.*Favorite*, contentDescription = "Favorite")  
 **}  
 }  
 }  
 }  
 }**}  
  
@Composable  
fun ComplexCard2(onAddToCart: () -> Unit, modifier: Modifier = Modifier) {  
 Card(  
 modifier = modifier  
 .*padding*(16.*dp*)  
 .*fillMaxWidth*(),  
 shape = *RoundedCornerShape*(16.*dp*),  
 elevation = CardDefaults.cardElevation(8.*dp*)  
 ) **{** Column **{** val image: Painter = painterResource(id = R.drawable.*burger2*)  
 Image(  
 painter = image,  
 contentDescription = null,  
 contentScale = ContentScale.Crop,  
 modifier = Modifier  
 .*height*(300.*dp*)  
 .*fillMaxWidth*()  
 )  
 Column(modifier = Modifier.*padding*(16.*dp*)) **{** Text(text = "Card1 Title1", fontSize = 20.*sp*, fontWeight = FontWeight.Bold)  
 Spacer(modifier = Modifier.*height*(8.*dp*))  
 Text(text = "Card1 Subtitle1", fontSize = 16.*sp*, color = Color.Gray)  
 Spacer(modifier = Modifier.*height*(8.*dp*))  
 Text(  
 text = "This is a description for the card1. It can be multiple lines long and should give a brief overview of the content.",  
 fontSize = 14.*sp* )  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
 Row(  
 verticalAlignment = Alignment.CenterVertically  
 ) **{** Button(onClick = onAddToCart) **{** Text("Add to cart")  
 **}** Spacer(modifier = Modifier.*width*(8.*dp*))  
 IconButton(onClick = **{** /\* *TODO* \*/ **}**) **{** Icon(Icons.Filled.*Favorite*, contentDescription = "Favorite")  
 **}  
 }  
 }  
 }  
 }**}  
  
@Composable  
fun ComplexCard3(onAddToCart: () -> Unit, modifier: Modifier = Modifier) {  
 Card(  
 modifier = modifier  
 .*padding*(16.*dp*)  
 .*fillMaxWidth*(),  
 shape = *RoundedCornerShape*(16.*dp*),  
 elevation = CardDefaults.cardElevation(8.*dp*)  
 ) **{** Column **{** val image: Painter = painterResource(id = R.drawable.*\_d\_burger*)  
 Image(  
 painter = image,  
 contentDescription = null,  
 contentScale = ContentScale.Crop,  
 modifier = Modifier  
 .*height*(300.*dp*)  
 .*fillMaxWidth*()  
 )  
 Column(modifier = Modifier.*padding*(16.*dp*)) **{** Text(text = "Card1 Title1", fontSize = 20.*sp*, fontWeight = FontWeight.Bold)  
 Spacer(modifier = Modifier.*height*(8.*dp*))  
 Text(text = "Card1 Subtitle1", fontSize = 16.*sp*, color = Color.Gray)  
 Spacer(modifier = Modifier.*height*(8.*dp*))  
 Text(  
 text = "This is a description for the card1. It can be multiple lines long and should give a brief overview of the content.",  
 fontSize = 14.*sp* )  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
 Row(  
 verticalAlignment = Alignment.CenterVertically  
 ) **{** Button(onClick = onAddToCart) **{** Text("Add to cart")  
 **}** Spacer(modifier = Modifier.*width*(8.*dp*))  
 IconButton(onClick = **{** /\* *TODO* \*/ **}**) **{** Icon(Icons.Filled.*Favorite*, contentDescription = "Favorite")  
 **}  
 }  
 }  
 }  
 }**}

**screens/CartViewModel.kt**

package com.example.appbarcenteraligned.ui.screens  
  
import androidx.lifecycle.ViewModel  
import androidx.lifecycle.*viewModelScope*import kotlinx.coroutines.flow.MutableStateFlow  
import kotlinx.coroutines.flow.StateFlow  
import kotlinx.coroutines.launch  
import com.example.appbarcenteraligned.R  
  
data class CartItem(val title: String, val subtitle: String, val imageRes: Int)  
  
class CartViewModel : ViewModel() {  
 private val \_cartItems = *MutableStateFlow*<List<CartItem>>(*emptyList*())  
 val cartItems: StateFlow<List<CartItem>> = \_cartItems  
  
 fun addItem(item: CartItem) {  
 *viewModelScope*.*launch* **{** \_cartItems.value = \_cartItems.value + item  
 **}** }  
  
 fun removeItem(item: CartItem) {  
 *viewModelScope*.*launch* **{** \_cartItems.value = \_cartItems.value - item  
 **}** }  
}

**screens/ScreenLogin.kt**

package com.example.appbarcenteraligned.ui.screens  
  
import androidx.compose.foundation.layout.\*  
import androidx.compose.material.icons.Icons  
import androidx.compose.material.icons.filled.*ArrowBack*import androidx.compose.material3.\*  
import androidx.compose.runtime.Composable  
import androidx.compose.ui.Modifier  
import androidx.compose.ui.Alignment  
import androidx.navigation.NavHostController  
  
@OptIn(ExperimentalMaterial3Api::class)  
@Composable  
fun ScreenLogin(navController: NavHostController) {  
 Scaffold(  
 topBar = **{** CenterAlignedTopAppBar(  
 title = **{** Text("Login") **}**,  
 navigationIcon = **{** IconButton(onClick = **{** navController.popBackStack() **}**) **{** Icon(Icons.Filled.*ArrowBack*, contentDescription = "Back")  
 **}  
 }** )  
 **}** ) **{** innerPadding **->** Column(  
 modifier = Modifier  
 .*padding*(innerPadding)  
 .*fillMaxSize*(),  
 horizontalAlignment = Alignment.CenterHorizontally,  
 verticalArrangement = Arrangement.Center  
 ) **{** Text("This is the Login screen")  
 **}  
 }**}

**screens/ScreenMessage.kt**

package com.example.appbarcenteraligned.ui.screens  
  
import androidx.compose.foundation.layout.\*  
import androidx.compose.material.icons.Icons  
import androidx.compose.material.icons.filled.*ArrowBack*import androidx.compose.material3.\*  
import androidx.compose.runtime.Composable  
import androidx.compose.ui.Modifier  
import androidx.compose.ui.Alignment  
import androidx.navigation.NavHostController  
  
@OptIn(ExperimentalMaterial3Api::class)  
@Composable  
fun ScreenMessage(navController: NavHostController) {  
 Scaffold(  
 topBar = **{** CenterAlignedTopAppBar(  
 title = **{** Text("Message") **}**,  
 navigationIcon = **{** IconButton(onClick = **{** navController.popBackStack() **}**) **{** Icon(Icons.Filled.*ArrowBack*, contentDescription = "Back")  
 **}  
 }** )  
 **}** ) **{** innerPadding **->** Column(  
 modifier = Modifier  
 .*padding*(innerPadding)  
 .*fillMaxSize*(),  
 horizontalAlignment = Alignment.CenterHorizontally,  
 verticalArrangement = Arrangement.Center  
 ) **{** Text("This is the Message screen")  
 **}  
 }**}

**screens/ScreenRateUs.kt**

package com.example.appbarcenteraligned.ui.screens  
  
import androidx.compose.foundation.layout.\*  
import androidx.compose.material.icons.Icons  
import androidx.compose.material.icons.filled.*ArrowBack*import androidx.compose.material3.\*  
import androidx.compose.runtime.Composable  
import androidx.compose.ui.Modifier  
import androidx.compose.ui.Alignment  
import androidx.navigation.NavHostController  
  
@OptIn(ExperimentalMaterial3Api::class)  
@Composable  
fun ScreenRateUs(navController: NavHostController) {  
 Scaffold(  
 topBar = **{** CenterAlignedTopAppBar(  
 title = **{** Text("Rate Us") **}**,  
 navigationIcon = **{** IconButton(onClick = **{** navController.popBackStack() **}**) **{** Icon(Icons.Filled.*ArrowBack*, contentDescription = "Back")  
 **}  
 }** )  
 **}** ) **{** innerPadding **->** Column(  
 modifier = Modifier  
 .*padding*(innerPadding)  
 .*fillMaxSize*(),  
 horizontalAlignment = Alignment.CenterHorizontally,  
 verticalArrangement = Arrangement.Center  
 ) **{** Text("This is the Rate Us screen")  
 **}  
 }**}

**screens/ScreenSettings.kt**

package com.example.appbarcenteraligned.ui.screens  
  
import androidx.compose.foundation.layout.\*  
import androidx.compose.material.icons.Icons  
import androidx.compose.material.icons.filled.*ArrowBack*import androidx.compose.material3.\*  
import androidx.compose.runtime.Composable  
import androidx.compose.ui.Modifier  
import androidx.compose.ui.Alignment  
import androidx.navigation.NavHostController  
  
@OptIn(ExperimentalMaterial3Api::class)  
@Composable  
fun ScreenSettings(navController: NavHostController) {  
 Scaffold(  
 topBar = **{** CenterAlignedTopAppBar(  
 title = **{** Text("Settings") **}**,  
 navigationIcon = **{** IconButton(onClick = **{** navController.popBackStack() **}**) **{** Icon(Icons.Filled.*ArrowBack*, contentDescription = "Back")  
 **}  
 }** )  
 **}** ) **{** innerPadding **->** Column(  
 modifier = Modifier  
 .*padding*(innerPadding)  
 .*fillMaxSize*(),  
 horizontalAlignment = Alignment.CenterHorizontally,  
 verticalArrangement = Arrangement.Center  
 ) **{** Text("This is the Settings screen")  
 **}  
 }**}

**screens/ScreenShare.kt**

package com.example.appbarcenteraligned.ui.screens  
  
import androidx.compose.foundation.layout.\*  
import androidx.compose.material.icons.Icons  
import androidx.compose.material.icons.filled.*ArrowBack*import androidx.compose.material3.\*  
import androidx.compose.runtime.Composable  
import androidx.compose.ui.Modifier  
import androidx.compose.ui.Alignment  
import androidx.navigation.NavHostController  
  
@OptIn(ExperimentalMaterial3Api::class)  
@Composable  
fun ScreenShare(navController: NavHostController) {  
 Scaffold(  
 topBar = **{** CenterAlignedTopAppBar(  
 title = **{** Text("Share") **}**,  
 navigationIcon = **{** IconButton(onClick = **{** navController.popBackStack() **}**) **{** Icon(Icons.Filled.*ArrowBack*, contentDescription = "Back")  
 **}  
 }** )  
 **}** ) **{** innerPadding **->** Column(  
 modifier = Modifier  
 .*padding*(innerPadding)  
 .*fillMaxSize*(),  
 horizontalAlignment = Alignment.CenterHorizontally,  
 verticalArrangement = Arrangement.Center  
 ) **{** Text("This is the Share screen")  
 **}  
 }**}

**screens/ScreenSync.kt**

package com.example.appbarcenteraligned.ui.screens  
  
import androidx.compose.foundation.layout.\*  
import androidx.compose.material.icons.Icons  
import androidx.compose.material.icons.filled.*ArrowBack*import androidx.compose.material3.\*  
import androidx.compose.runtime.Composable  
import androidx.compose.ui.Modifier  
import androidx.compose.ui.Alignment  
import androidx.navigation.NavHostController  
  
@OptIn(ExperimentalMaterial3Api::class)  
@Composable  
fun ScreenSync(navController: NavHostController) {  
 Scaffold(  
 topBar = **{** CenterAlignedTopAppBar(  
 title = **{** Text("Sync") **}**,  
 navigationIcon = **{** IconButton(onClick = **{** navController.popBackStack() **}**) **{** Icon(Icons.Filled.*ArrowBack*, contentDescription = "Back")  
 **}  
 }** )  
 **}** ) **{** innerPadding **->** Column(  
 modifier = Modifier  
 .*padding*(innerPadding)  
 .*fillMaxSize*(),  
 horizontalAlignment = Alignment.CenterHorizontally,  
 verticalArrangement = Arrangement.Center  
 ) **{** Text("This is the Sync screen")  
 **}  
 }**}

**screens/ScreenTrash.kt**

package com.example.appbarcenteraligned.ui.screens  
  
import androidx.compose.foundation.layout.\*  
import androidx.compose.material.icons.Icons  
import androidx.compose.material.icons.filled.*ArrowBack*import androidx.compose.material3.\*  
import androidx.compose.runtime.Composable  
import androidx.compose.ui.Modifier  
import androidx.compose.ui.Alignment  
import androidx.navigation.NavHostController  
  
@OptIn(ExperimentalMaterial3Api::class)  
@Composable  
fun ScreenTrash(navController: NavHostController) {  
 Scaffold(  
 topBar = **{** CenterAlignedTopAppBar(  
 title = **{** Text("Trash") **}**,  
 navigationIcon = **{** IconButton(onClick = **{** navController.popBackStack() **}**) **{** Icon(Icons.Filled.*ArrowBack*, contentDescription = "Back")  
 **}  
 }** )  
 **}** ) **{** innerPadding **->** Column(  
 modifier = Modifier  
 .*padding*(innerPadding)  
 .*fillMaxSize*(),  
 horizontalAlignment = Alignment.CenterHorizontally,  
 verticalArrangement = Arrangement.Center  
 ) **{** Text("This is the Trash screen")  
 **}  
 }**}

**utils/DrawItem.kt**

package com.example.appbarcenteraligned.utils  
  
import androidx.compose.foundation.clickable  
import androidx.compose.foundation.layout.\*  
import androidx.compose.material3.Icon  
import androidx.compose.material3.Text  
import androidx.compose.runtime.Composable  
import androidx.compose.ui.Modifier  
import androidx.compose.ui.graphics.vector.ImageVector  
import androidx.compose.ui.Alignment  
import androidx.compose.ui.unit.dp  
  
@Composable  
fun DrawerItem(icon: ImageVector, label: String, onClick: () -> Unit) {  
 Row(  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*clickable*(onClick = onClick)  
 .*padding*(16.*dp*),  
 verticalAlignment = Alignment.CenterVertically  
 ) **{** Icon(icon, contentDescription = label)  
 Spacer(modifier = Modifier.*width*(16.*dp*))  
 Text(label)  
 **}**}

**components/AdvancedAlertDialog.kt**

package com.example.appbarcenteraligned.ui.components  
  
import androidx.compose.foundation.Image  
import androidx.compose.foundation.layout.\*  
import androidx.compose.material3.AlertDialog  
import androidx.compose.material3.Text  
import androidx.compose.material3.TextButton  
import androidx.compose.runtime.Composable  
import androidx.compose.runtime.MutableState  
import androidx.compose.ui.Modifier  
import androidx.compose.ui.graphics.painter.Painter  
import androidx.compose.ui.unit.dp  
  
@Composable  
fun AdvancedAlertDialog(  
 showDialog: MutableState<Boolean>,  
 onNavigateToScreen1: () -> Unit,  
 onNavigateToScreen2: () -> Unit,  
 imagePainter: Painter  
) {  
 AlertDialog(  
 onDismissRequest = **{** showDialog.value = false **}**,  
 title = **{** Text("Floating Button Pressed") **}**,  
 text = **{** Column **{** Image(  
 painter = imagePainter,  
 contentDescription = null,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*height*(150.*dp*)  
 .*padding*(bottom = 16.*dp*)  
 )  
 Text("You pressed on the Floating button")  
 **}  
 }**,  
 confirmButton = **{** Row(  
 modifier = Modifier.*fillMaxWidth*(),  
 horizontalArrangement = Arrangement.SpaceBetween  
 ) **{** TextButton(onClick = **{** showDialog.value = false  
 onNavigateToScreen1()  
 **}**) **{** Text("Screen 1")  
 **}** TextButton(onClick = **{** showDialog.value = false  
 onNavigateToScreen2()  
 **}**) **{** Text("Screen 2")  
 **}  
 }  
 }** )  
}

**components/BottomSheetContent.kt**

package com.example.appbarcenteraligned.ui.components  
  
import androidx.compose.foundation.Image  
import androidx.compose.foundation.layout.\*  
import androidx.compose.foundation.shape.RoundedCornerShape  
import androidx.compose.material.\*  
import androidx.compose.runtime.\*  
import androidx.compose.ui.Alignment  
import androidx.compose.ui.Modifier  
import androidx.compose.ui.draw.clip  
import androidx.compose.ui.graphics.Color  
import androidx.compose.ui.res.painterResource  
import androidx.compose.ui.unit.dp  
import com.example.appbarcenteraligned.R  
import kotlinx.coroutines.launch  
  
@OptIn(ExperimentalMaterialApi::class)  
@Composable  
fun BottomSheetContent(onHide: () -> Unit) {  
 Box(modifier = Modifier.*fillMaxSize*()) **{** Column(  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*padding*(16.*dp*)  
 ) **{** Text("Bottom Sheet Content", style = MaterialTheme.typography.h6)  
 Spacer(modifier = Modifier.*height*(8.*dp*))  
 Image(  
 painter = painterResource(id = R.drawable.*burger2*),  
 contentDescription = "Example Image",  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*height*(100.*dp*)  
 .*clip*(*RoundedCornerShape*(8.*dp*))  
 )  
 Spacer(modifier = Modifier.*height*(8.*dp*))  
 Card(  
 shape = *RoundedCornerShape*(8.*dp*),  
 modifier = Modifier.*fillMaxWidth*(),  
 backgroundColor = Color.LightGray  
 ) **{** Column(modifier = Modifier.*padding*(16.*dp*)) **{** Text("Card Title", style = MaterialTheme.typography.h6)  
 Spacer(modifier = Modifier.*height*(8.*dp*))  
 Text("Card content goes here. You can place additional text, images, or other components inside a card.")  
 **}  
 }  
 }** Button(  
 onClick = onHide,  
 modifier = Modifier  
 .*align*(Alignment.TopEnd)  
 .*padding*(16.*dp*)  
 ) **{** Text("Hide")  
 **}  
 }**}

**components/DrawerContent.kt**

package com.example.appbarcenteraligned.ui.components  
  
import androidx.compose.foundation.layout.\*  
import androidx.compose.material.icons.Icons  
import androidx.compose.material.icons.filled.\*  
import androidx.compose.material3.\*  
import androidx.compose.runtime.Composable  
import androidx.compose.ui.Modifier  
import androidx.compose.ui.unit.dp  
import androidx.navigation.NavHostController  
import com.example.appbarcenteraligned.utils.DrawerItem  
import kotlinx.coroutines.CoroutineScope  
import kotlinx.coroutines.launch  
  
@Composable  
fun DrawerContent(navController: NavHostController, drawerState: DrawerState, coroutineScope: CoroutineScope) {  
 ModalDrawerSheet **{** Text("Drawer title", modifier = Modifier.*padding*(16.*dp*))  
 Divider()  
 DrawerItem(icon = Icons.Filled.*Home*, label = "Screen 1", onClick = **{** coroutineScope.*launch* **{** drawerState.close() **}** navController.navigate("screen1")  
 **}**)  
 DrawerItem(icon = Icons.Filled.*MailOutline*, label = "Screen 2", onClick = **{** coroutineScope.*launch* **{** drawerState.close() **}** navController.navigate("screen2")  
 **}**)  
 DrawerItem(icon = Icons.Filled.*Star*, label = "Sync", onClick = **{** coroutineScope.*launch* **{** drawerState.close() **}** navController.navigate("sync")  
 **}**)  
 DrawerItem(icon = Icons.Filled.*Delete*, label = "Trash", onClick = **{** coroutineScope.*launch* **{** drawerState.close() **}** navController.navigate("trash")  
 **}**)  
 DrawerItem(icon = Icons.Filled.*Settings*, label = "Settings", onClick = **{** coroutineScope.*launch* **{** drawerState.close() **}** navController.navigate("settings")  
 **}**)  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
 Text("Profile", modifier = Modifier.*padding*(start = 16.*dp*))  
 DrawerItem(icon = Icons.Filled.*AccountBox*, label = "Login", onClick = **{** coroutineScope.*launch* **{** drawerState.close() **}** navController.navigate("login")  
 **}**)  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
 Text("Communicate", modifier = Modifier.*padding*(start = 16.*dp*))  
 DrawerItem(icon = Icons.Filled.*Share*, label = "Share", onClick = **{** coroutineScope.*launch* **{** drawerState.close() **}** navController.navigate("share")  
 **}**)  
 DrawerItem(icon = Icons.Filled.*ThumbUp*, label = "Rate us", onClick = **{** coroutineScope.*launch* **{** drawerState.close() **}** navController.navigate("rate\_us")  
 **}**)  
 **}**}

**components/MyScrollContent.kt**

package com.example.appbarcenteraligned.ui.components  
  
import androidx.compose.foundation.layout.\*  
import androidx.compose.foundation.rememberScrollState  
import androidx.compose.foundation.verticalScroll  
import androidx.compose.material3.MaterialTheme  
import androidx.compose.material3.Text  
import androidx.compose.runtime.Composable  
import androidx.compose.ui.Modifier  
import androidx.compose.ui.unit.dp  
  
@Composable  
fun MyScrollContent(padding: PaddingValues) {  
 val scrollState = rememberScrollState()  
 Column(  
 modifier = Modifier  
 .*padding*(padding)  
 .*verticalScroll*(scrollState)  
 .*fillMaxSize*()  
 ) **{** *repeat*(20) **{** index **->** Text(  
 text = "Item $index",  
 modifier = Modifier  
 .*padding*(16.*dp*)  
 .*fillMaxWidth*()  
 )  
 **}  
 }**}

**Sample 10**

**Sample 5**